PREMIER ISSUE

$4.95

A PUBLICATION OF
THE HARVARD GROUP

SQ SYNTAX QUARTERLY

Serving The Timex-Sinclair Family of Personal Computers

Winter 1982 Vol. 1, No. 1

Build a Printer Interface and EPROM Programmer– Part 1

Big Keyboard for ZX/TS:
How to Buy, How to Build

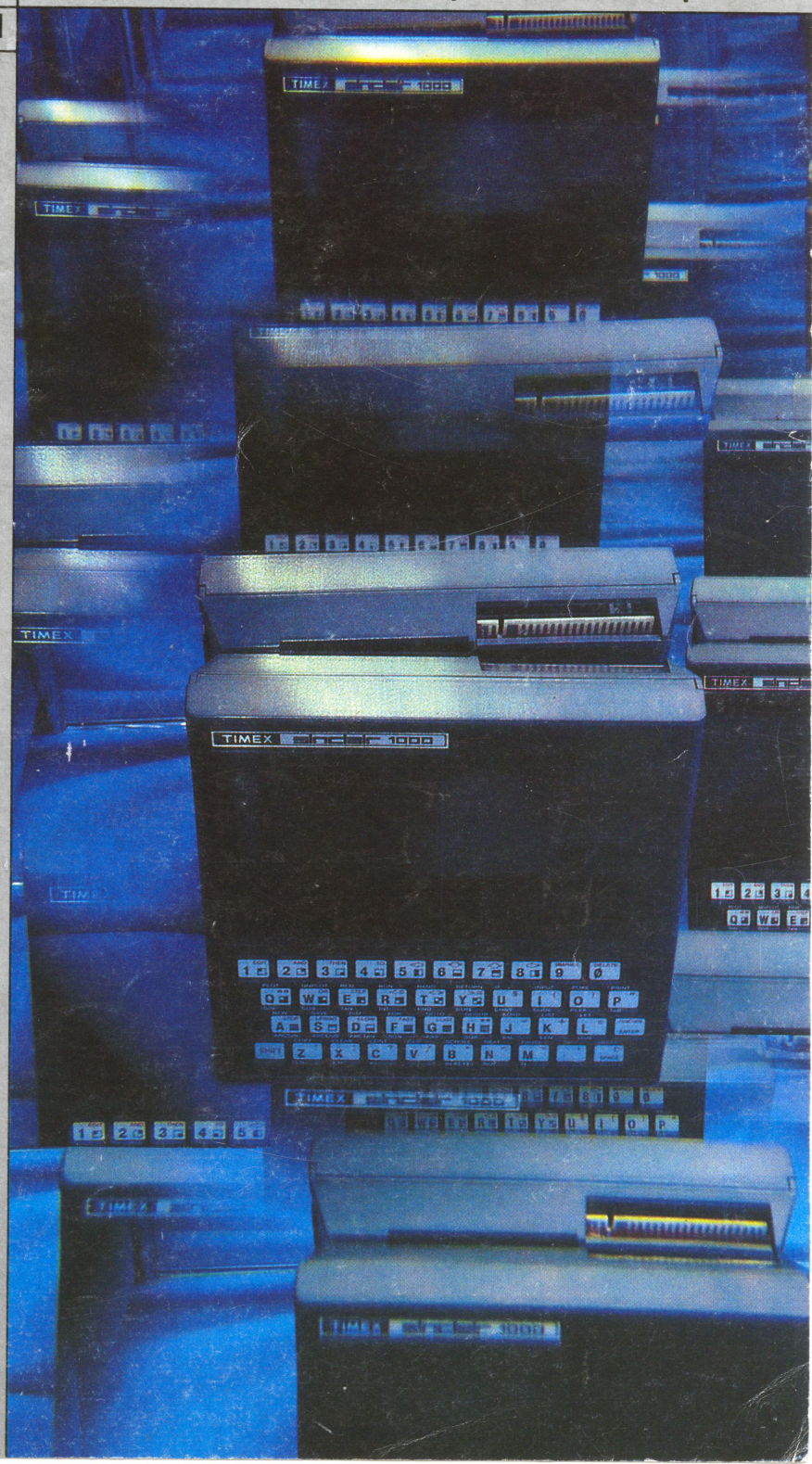
Machine Code Programs:
Where and How to Load Them

Beginning BASIC:
Communicating with Your Computer

ZX/TS Memory Map

ZX/TS Programs

New ZX/TS Software and Hardware Reviews



0 26632 12083 11

CHANGE YOUR

TIMEX/SINCLAIR 1000 DISPLAY

TO FULL COLOUR GRAPHICS

with

KOLORWORKS

LOOK AT THE FEATURES !!!

* Plugs into ZX81/1000 (NO SOLDERING NEEDED)
* All text will run on the color tv
* Module contains it's own memory
* User defined characters & graphics up to 256x192 pixels
* Latest technology with TMS9918 VDP
* Module contains extention of basic commands including: PAPER / INK / BORDER / BIN / SPRITE / OUT / INP / etc.

for $149.95

MAIL ORDERS STARTING DEC. 1st

Use your KOLORWORKS immediately with a game cassette for $9.95

EARLYBIRD SPECIAL - with orders placed between Dec.1 & Dec.31 recieve the game cassette ABSOLUTELY FREE

For further information

Send $2.00 tot (No phone info.)

BRAINCHILD COMPUTER WORKS,INC.
P.O. BOX 506
PEWAUKEE, WI. 53072

$2.00 Credited to your order

MAIL TO: Brainchild Computer Works, Inc.
P. O. Box 506
Pewaukee, WI 53072

	Price +	Qty.	Amount
KOLORWORKS	$ 149.95		
cassette	9.95		
Shipping and Handling	4.95		$4.95
		Total	

My ☐ check ☐ money order enclosed.

Name _____

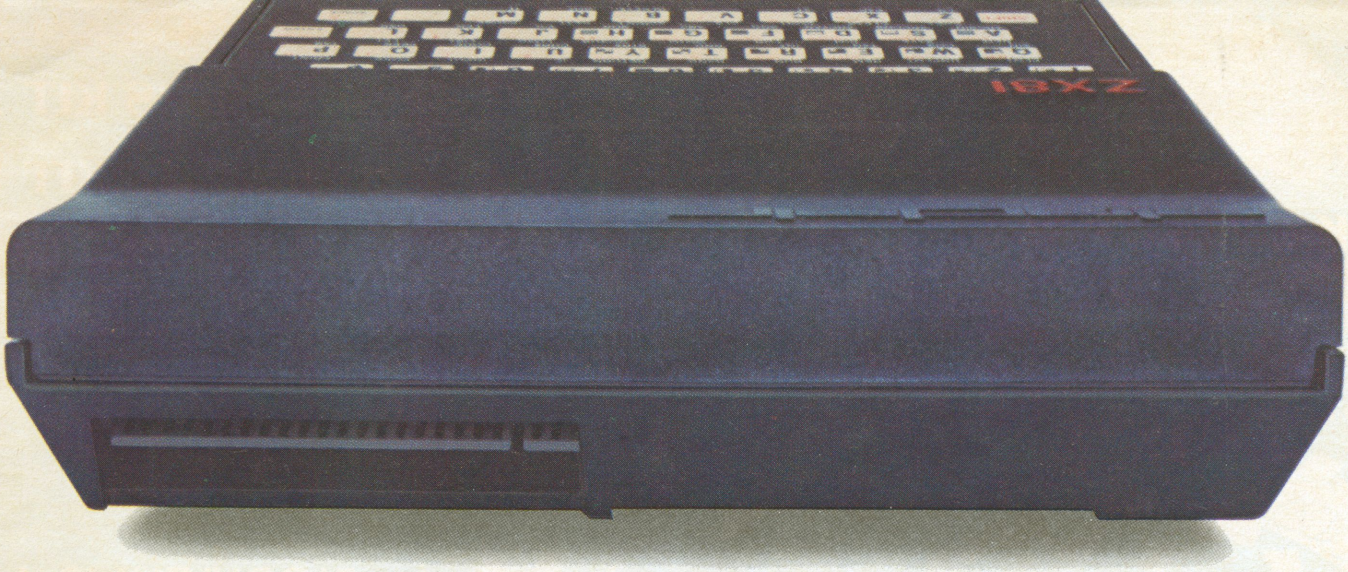
Street _____

City _____ **State** _____ **Zip** _____

Please allow six to eight weeks for processing. Thank you.

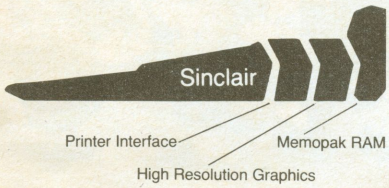
WATCH FOR:

"GAMMWORKS" - Joystick ports / sound / cartridge slot



BEHIND EVERY GOOD SINCLAIR IS A MEMOPAK

If you own a Timex-Sinclair 1000 or ZX81 computer, you should have a Memopak behind it. From increased memory to high resolution graphics, Memotech has a Memopak to boost your system's capabilities. Every Memopak peripheral comes in a black anodised aluminum case and is designed to fit together in "piggy back" fashion to enable you to continue to add on and still keep an integrated system look.



Order at no risk

All Memotech products carry our 10 day money back guarantee. If you're not completely satisfied, return it in ten days and we will give you a full refund.
And every Memotech product comes with a six month warranty. Should anything be defective with your Memopak, return it to us and we will repair or replace it free of charge. Dealer inquiries welcome. To order any Memotech product call our toll-free number **800/662-0949** or use the order coupon.

MEMOTECH
CORPORATION

7550 West Yale Avenue
Denver, Colorado 80227
(303) 986-1516
TWX 910-320-2917

Mail to: Memotech Corporation, 7550 West Yale Ave., Denver, CO 80227
Code: SQ1

	*Price	Qty.	Total
64K RAM	$179.95		
32K RAM	109.95		
16K RAM	59.95		
Centronics Parallel Printer Interface	104.95		
RS232 Printer Interface	139.95		
High Resolution Graphics	144.95		
Shipping and handling	4.95		$4.95
		Tax**	
		Total	

* All prices quoted in U.S. dollars
** Colorado residents please add sales tax
☐ Check ☐ MasterCard ☐ Visa

Account No. _____ Exp. _____

Name

Address

City State Zip

Memopak 64K RAM The 64K RAM extends the memory of your Sinclair by 56K to a full 64K. It is directly addressable, user transparent, is neither switched nor paged and accepts such BASIC commands as 10 DIM A (9000). The Memopak 64K turns your Sinclair into a powerful computer suitable for business, recreational and educational use. No additional power supply is required.

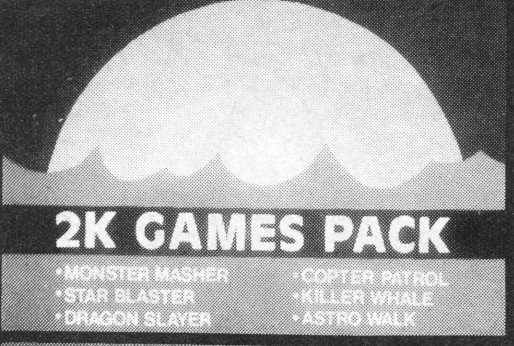
Memopak 32K RAM The 32K RAM Memopak offers your Sinclair a full 32K of directly addressable RAM. Like the 64K Memopak, it is neither switched nor paged and enables you to execute sophisticated programs and store large data bases. It is also fully compatible with Sinclair's or Memotech's 16K RAM to give you a full 48K of RAM.

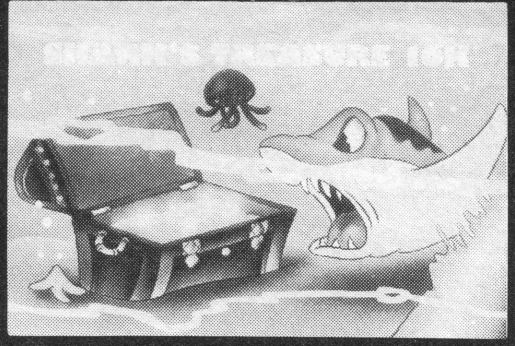
Memopak 16K RAM The Memopak 16K RAM provides an economical way to increase the capabilities of your Sinclair. And at the same time, it enables you to continue to add on other features with its "piggy back" connectors. It is compatible with the Sinclair 16K or a second Memopak 16K or Memopak 32K to give 32K or 48K of RAM respectively.

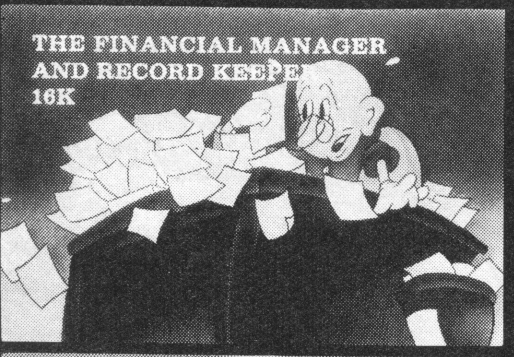
Memopak High Resolution Graphics The Memopak HRG contains a 2K EPROM monitor and is fully programmable for high resolution graphics. The HRG provides for up to 192 by 248 pixel resolution.

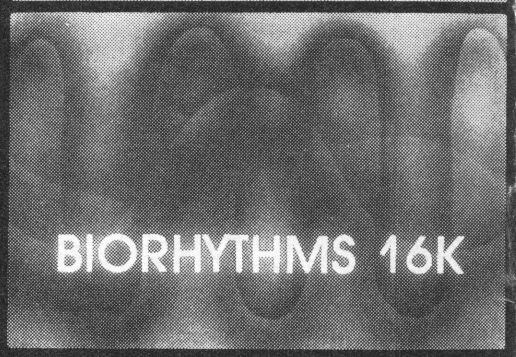
Memopak Printer Interface The Memopak Centronics Parallel or RS232 Interface paks enable your Sinclair to use a wide range of compatible printers (major manufacturers' printers available through Memotech at significant savings). The resident software in the units gives the ASCII set of characters. Both Memopak printer interfaces provide lower case character capabilities. The RS232 Interface is also compatible with modems.

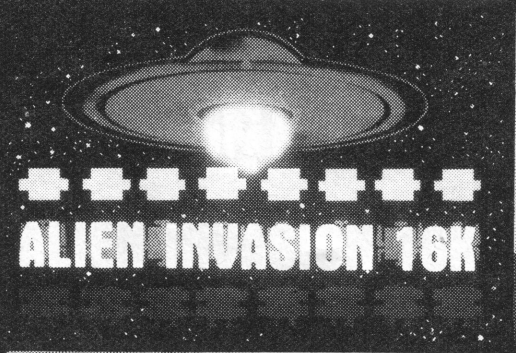
New products coming soon Memotech will soon be introducing four new Sinclair compatible products: a high quality, direct connection keyboard, a digitizing tablet, a 16K EPROM and a disk drive. Watch for our future advertisements.

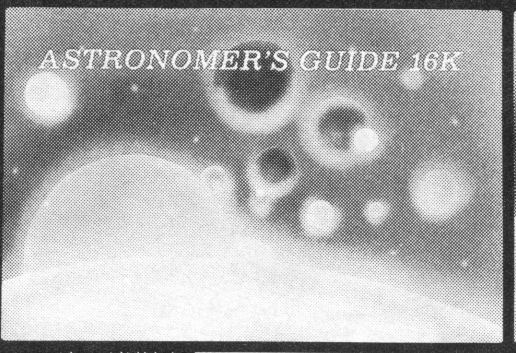


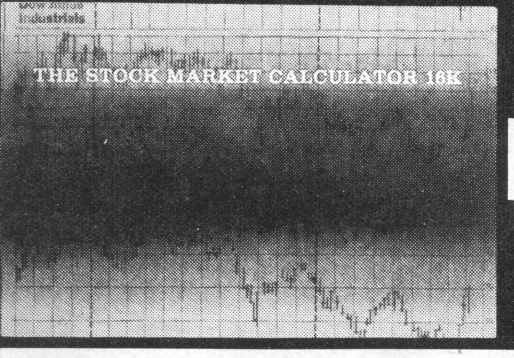












MORE TS1000/ZX81 SOFTWARE FROM...

SOFTSYNC, INC.

14 East 34th St. New York, N.Y. 10016

(212) 685-2080 M/C & VISA ACCEPTED

WRITE FOR FREE CATALOG & PROGRAM LISTING

PREMIER ISSUE

A PUBLICATION OF
THE HARVARD GROUP

SQ SYNTAX QUARTERLY

Serving The Timex-Sinclair Family of Personal Computers

Winter 1982 Vol. 1, No. 1

SQ Staff

Publisher
Kirtland H. Olson, P.E.

Editor
Ann Zevnik Doyle

Assistant Editor
Pamela Petrakos-Wilson

Technical Consultant
Eric K. Olson

Editorial Assistants
Pamela K. Vinal
Karen Brody

Production Manager
Erik W. Slickman

Circulation
Rosalie Adler
Constance Saarinen
Timothy Parker

Financial Manager
Mary Russo

Designer
RDS Designs

Cover Photography
Gary N. Slickman

SQ, Syntax Quarterly (ISSN 0734-0133) is copyright © 1982 and published by Syntax ZX80, Inc., a wholly owned subsidiary of The Harvard Group, RD 2, Box 457, Bolton Rd., Harvard, MA 01451. Subscription prices are $15 for one year (4 issues), single copies are $4.95. A second class permit pending. Postage paid at Harvard, MA 01451 and additional mailing offices. POSTMASTER: Send address corrections to The Harvard Group, RD 2, Box 457, Harvard, MA 01451.

Contents

A Program to Index Articles—8K ROM, 16K RAM 10
Organize your favorite publications. *by Andrew Q. Banta*

Improved Super Monzxer Game 12
Escape from man-eating Monzxers and other beasts. *by Robert Calhoun*

Syntactic Sum with Variables 16
Check for good loads from tape with this utility program. *by Donald J. Beck*

MC Programs—Where and How to Load Them 20
Three ways to load either decimal or hexadecimal machine code. *by William Wentz*

ZX/TS Directory Program 23
Your ZX/TS computer becomes an electronic address book. *by David R. Rowland*

Keyboard Conversions for ZX/TS 25
Hints on buying big keyboards and how to hook them up. *by David M. Straub*

ZX/TS Circuit Board 30
Find your way around inside your computer.

ZX80 Schematic 32

Build Your Own EPROM Programmer and Centronics Printer Interface 34
Part 1 of two parts details how to build your own expansion board and parallel printer port. *by John Oliger*

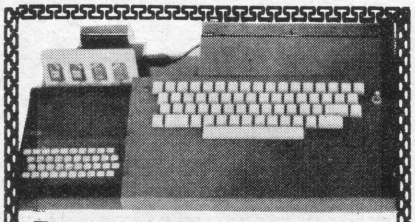
ZX81 Schematic 47

16K RAM Schematic 49

ZX Memory Map 51
Find out which peripherals work together and why. *by Stephen Adams*

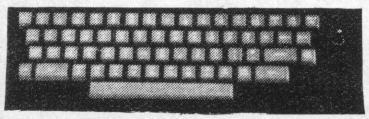
Departments

Beginners' BASIC 5
Software Reviews 58
Hardware Reviews 62
Ad Index 64



The SYNCHRONIZE SYSTEM

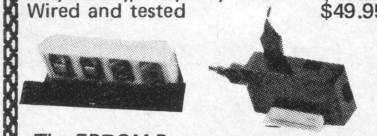
The Synchronize system is designed to put maximum fun and utility in your hands at minimum cost. Start with our Keyboard manual at $6.00. This manual covers what you need to know about keyboards in general, ours and theirs, and how to wire it to your Sinclair or Timex computer. It includes circuits on how to use extra keys for Shift-lock, Auto-repeat, and other user-defined functions, plans for a 3 amp power supply that let you run your computer and add-ons most anywhere, including your car, and plans for a case with tape and/or module storage. If you decide to buy a keyboard later, you may deduct the price of the manual. Manuals for our other products may also be purchased separately and later deducted from the price of the kit or wired module.



This keyboard was originally made for T.I. It has 62 full-sized keys with gold contacts mounted on a metal plate 4" by 15.5". $37.50

The Joystick Controller handles up to 4 Atari type joysticks. It allows full 8 direction movement and fire control in machine code for fast response.

Manual	$ 6.00
Bare-board No manual	$12.00
Kit; board,parts,case,manual	$39.95
Wired and tested	$49.95



The EPROM Programmer puts an end to cassette headaches by letting you store programs in ROM. Simply load a program, then your computer can store it in one or more 2716 EPROM's. The module holds four 2716's, giving you 8K of ROM in the 8K to 16K slot. By using ZIF (zero insertion force) sockets, ROM's can be changed as easily as cassettes.

Manual	$ 6.00
Bare-board No manual	$19.75
Semi-Kit	$49.85
board, parts, except EPROM's, plain sockets, manual, No case.	
Full-Kit; includes ZIF sockets and case. Without EPROM's	$74.45
With 4 EPROM's	$94.45
Wired and tested; includes ZIF sockets	
Without EPROM's	$86.95
With 4 EPROM's	$106.95

All prices postpaid U.S.A. only. In Canada add $2.00 for manuals or $5.00 for all other articles. All other countries add $4.00 for manuals or $10.00 for all other articals. U.S.A. funds only. Check,money order,VISA, Mastercharge. Phone orders accepted. Texas residents add 5% state sales tax.

SYNCHRONIZE
P.O. Box 1667
Kerrville, Tx. 78028
1-512-896-1288

This Could be Your Chance for Fame and Fortune

(or at least fame)

Would you like to see your name in print? **SQ** and **SYNTAX** newsletter are looking for authors. If you have a program or article you'd like to share with other ZX/TS owners, send it along.

Our requirements are simple: all we ask is that your program run with no bugs and your article present accurate information.

However, here are a few more details that enhance your chances of being published:

PROGRAMS: Any program for a fun or useful purpose is a candidate for **SYNTAX** or **SQ**. We publish game, business, educational, utility and science/math programs. Please send programs on cassette, particularly if they're long. We print programs directly from a Sinclair printer to minimize errors. If you send your program on cassette, it will appear exactly as you entered it. A hard, or paper, copy helps us figure out any problems that may arise. Also include as much documentation as you can, including a list of variables used and their functions. Write complete instructions for using your program so the most inexperienced user could not mess it up. Sample output also helps, especially if your program generates charts, graphs, or numerical tables. Make your program as user-friendly as possible within the limitations of available memory. Indicate how much memory your program requires. And include the Syntactic Sum of the program as listed. For a copy of the Syntactic Sum program, contact SYNTAX (address following).

ARTICLES: Articles can cover virtually any topic related to ZX/TS computers and their applications. Our readers run the gamut from expert to novice, so we are looking for articles written at all levels of complexity. Your article can present a programming tip, explain the function of the machine or a command, or describe a hardware project. If you present ideas on programming, it helps to include short sample routines demonstrating your ideas. If your article discusses hardware, please include sketches, schematics, or even photos to help readers follow along.

For both programs and articles, type your submission or print clearly, preferably double-spaced. Put your name, address and day-time phone number on the first page. If you want us to return your material to you, include a self-addressed stamped envelope.

You can submit to either **SYNTAX** newsletter or **SQ** magazine. Generally, programs and articles in **SQ** tend to be longer and more involved than those in **SYNTAX** because of space considerations. Which publication your submission ultimately appears in is up to the editors.

What's in it for you? Aside from the fame and glory of appearing in **SYNTAX** or **SQ**, you'll receive a check from us dependent on the length of the published material. We pay 7 cents for each 6 characters, including spaces and punctuation. This payment buys us the nonexclusive right to use your material. We can use it and so can you, so you can sell it again to anyone you like. Programs receive a token $2 payment each. Not enough to retire on, but often enough to pay for your subscription.

So dig through your notes and files and send off anything you think may help other users use their computers better. Address submissions to:

Editor
SYNTAX/SQ
Rd 2, Box 457
Harvard, MA 01451
USA

SINWARE provides high-quality machine-code programs for the ZX81 or TS1000.

Z EXTRA

Z EXTRA gives you fast direct control of graphics and display features from the keyboard. No programming required to create, save, print or display multiple screens. Just enter text and pictures and menu-select for great moving displays.

Z EXTRA features blinking cursors, repeating keys and many convenient functions for creating screen displays.

HOT Z is an easy-to-use disassembler/editor and machine-language monitor. Extensive debugging facilities let you track down errors fast. Cursor-driven editing simplifies corrections and revisions. Labels provide annotated listings that stay understandable.

Z EXTRA and **HOT Z** are each $19.95 on cassette and require a minimum of 16K. Please specify your equipment.

SINWARE
BOX 323, DIXON, NM 87527

CUSTOM-DESIGNED ATTACHE CASE
for
Sinclair ZX80/81
and
TS1000 Computers

Form fitted to hold:
Computer
16K Ram Module
Cassette Recorder
Power pack and cord
Protects your computer

SPECIAL $99.95 PPD
Check or money order

COMPUTA-DEK
6009 Kenwood Drive
Nashville, TN 37215

Getting Information into Your Computer

by James A. Conrad, Seattle, WA

Thoroughly understanding numeric and alphanumeric information, the variables that store information, and the assignment statements that put information into variables makes your programming quicker, easier and less frustrating. And I know very few programmers who couldn't improve their programs by writing tighter input checking routines. Even if you're a grizzled old pro at programming, a few minutes spent reviewing and experimenting with variables and assignment statements might improve your programming.

Kinds of Information

Think of your computer as a combination high-powered calculator and smart typewriter. It processes information—numbers like a calculator does, letters and words like a typewriter does. These two kinds of information are formally called *numeric* and *alphanumeric* information in computer lingo.

Most of us easily understand the idea of *numeric* information—numbers. We routinely add, subtract, multiply, divide, compute square roots and do a lot of other number manipulating with calculators.

Alphanumeric information is like typewriter characters—letters and numbers. In computer jargon we call this "string" information and we spell string "$." String information is designated by quotation marks, for example, "MIKE," "2," "SINCLAIR." When your computer sees the quotation marks, it knows it's dealing with strings.

Some novice programmers don't understand why an alphanumeric "2" can't be mathematically manipulated like a numeric 2. Remember alphanumeric characters are like typewriter characters—a "2" on a typewriter is just like another alphanumeric character, like "A" or "&" or "H" or "?." A typewriter doesn't recognize the character "2" as a number, nor does your computer when it's processing alphanumeric information.

Variables

Your computer stores numeric and alphanumeric information in memory cells called *variables*. Numeric variables hold numeric information and string variables contain alphanumeric information. Think of variables as little boxes in the computer's memory. When you're trying to figure out why a program is being ornery, draw some little variable boxes to show what each variable contains. Label each box and write in the information it contains.

Each variable has its own individual name. The first character of a numeric variable's name must be a letter, A to Z. Any characters after the first letter are optional and may be any combination of letters and numbers. A few examples are A, J222, SALES82.

String variables, on the other hand, can be named only by a single letter and must have the "$" string sign following the variable name. So, although we can have a large number of numeric variables, we can have only 26 string variables—A$ through Z$, one for each letter of the alphabet. String variable A$ is pronounced "A string." (When my son was learning about string variables, he continually referred to "X dollar sign"—and I continually corrected him to say "X string." I knew I had won when he asked, "Dad, may I borrow five strings?")*

Novices sometimes get confused by similarly named variables, such as A, A1, AA, and A$. Each of these is a different variable, related by similarity of name only, and each contains different information.

On checking with the author's son we found that he never repaid the $5. We question who really won. —Ed.

Assignment

An *assignment statement* sets the value of the variable named on the left equal to the value of the expression on the right of the equal sign. The most common is the LET statement of the form:

LET variable name = expression

The *expression* can be a constant, another variable, or a formula. Some examples:

LET A = 12	(constant)
LET B = A	(variable)
LET A6 = L*W	(formula)
LET G$ = "STRIPPER"	(constant)
LET P$ = N$	(variable)
LET C$ = R$ + "JOHN"	(formula)

When the expression is a formula, the LET statement calculates its value before assigning it to the variable.

The equal sign in an assignment statement looks like the same equal sign we remember from Algebra I—but it isn't. Assignment statements are not algebraic equations. In computer language, not all equal signs are created equal. (As you learn more about BASIC you'll find another use for the equal sign as a relational operator testing the truth of an equation. This is generally found in IF-THEN statements.)

Think of the equal sign in a LET statement as a left-pointing arrow (as in computer pseudocode). This arrow shows assignment (LET A ← 12). This reminds us that an assignment statement doesn't use an arithmetic equal sign, one that shows equality. Instead it assigns the value on the right to the variable on the left and so functions as an assignment sign.

Replacement

Novice programmers often encounter another algebraic enigma when they

assign a new value to an old variable. When I first saw the algebraic aberration K=K+1, I instinctively knew it was wrong. My old junior high algebra teacher's stern apparition appeared in my head and said, "Once you've given it a value, a variable is defined and you can't change it."

Not so. We're discussing computers, not algebra—assignment statements, not equations. When we assign a value to a variable, *the new value replaces the old*. Use the little boxes mentioned earlier to illustrate this concept.

	X is Now
10 LET X=2	2
20 LET X=X+2	4
30 LET X=X*X	16
40 LET X=X-6	10

When we assign the value of one variable to another (for example, LET B=A), what happens to the contents of the variable on the right? It remains unchanged—copied, not transferred, into the variable on the left.

Input Statements

LET statements are program-dependent: the data to be assigned are written into the program. To write an interactive program (one in which the user gives information to the computer), we need a *program-independent* assignment statement.

INPUT statements give us a way to enter data from the keyboard and assign them to variables. INPUT statements take this form:

INPUT variable

When carrying out an INPUT statement, the computer does three things:

1. Halts program execution.
2. Prints an [L] cursor in reverse video on the message line at the bottom of the screen and waits for you to enter data.
3. Assigns the data to the variable which follows the INPUT statement.

Run this quick program with various inputs:

```
10 INPUT R
20 PRINT R
```

What happens? Line 10 prints an L cursor on the message line and the program waits for input. When you enter numeric information, the INPUT statement assigns it to variable R. Line 20 prints it.

Try entering your name in response to the prompt. The message line shows a report code 2 (undefined variable) in line 10. Variable R is a numeric variable and you attempted to enter string (alphanumeric) information. But that's not what the report code is saying. It's telling you that your name is an undefined variable. Think about this as you try our next program.

What if we have a value in another variable, B for example, and want to copy it into variable R? Can we do this with an INPUT statement as we can with a LET statement? Try it:

```
10 PRINT B=17
20 INPUT R
30 PRINT R
```

When line 20 causes the L cursor to appear on the message line, try entering the letter B. The computer accepts it and prints the result, 17. In line 20 we alerted the computer to expect a numeric value. So when it sees B as input, it copies the numeric information from variable cell B into variable cell R. Line 30 prints it out.

Now do you see what happened in the previous program? The computer was looking for a number. When you entered your name it looked for a numeric value with your name on it. When it couldn't find one, it gave you report code 2. Try changing line 10 in the last program to LET your name = 17. Now enter your name in response to the prompt. It works.

Let's try assigning information to a string variable:

```
10 INPUT R$
20 PRINT R$
```

When we use a LET statement, we enclose string information in quotation marks. When you run this short program, notice that the L cursor on the message line now has quotation marks around it. The computer puts the quotation marks in for you! (If you don't believe it, try adding your own. You get a syntax error marker—a reverse video S in the message line.)

Run this several times trying various inputs, including graphics characters It works for almost everything except the quotation mark. (If you have to input a quotation mark, use the double quotation mark over the Q key.) The computer accepts and assigns the strings to the string variable R$.

What happens if we try to input numeric information into a string variable? Try it. No error message. The numbers are alphanumeric information—characters—and the computer accepts them. But it accepts them as strings, not as numbers.

Prompt Messages

Innocent users, confronted with an L cursor on the message line, probably won't understand that they are expected to enter something. So to make our programs user-friendly, we include messages telling the user what to enter:

```
10 PRINT "ENTER YOUR NAME"
20 INPUT N$
30 PRINT "ENTER A NUMBER, ";N$
40 INPUT R
50 PRINT "THE NUMBER YOU ENTER
ED IS ";R
```

Lines 10 and 30 are prompt messages, requesting the user to enter the desired information. Every INPUT statement should have a prompt message.

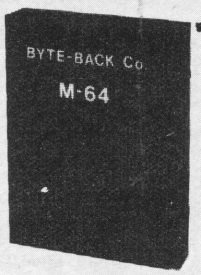
Input Checking

Well written programs are user-proof as well as user-friendly. Experienced programmers include input-checking routines with virtually all their INPUT statements. Here are two of the most useful.

The first checks numeric input. If the input is incorrect, the routine prints an error message and returns to the INPUT statement to allow you to re-enter data. (The <> sign means "not equal to." Type it into your computer by pressing the shift and T keys.)

BYTE-BACK modules

64-K MEMORY $119.95 KIT



WIRED and TESTED $129.95

IN STOCK!
SAME DAY SHIPMENT!

WHY PAY MORE?

BYTE-BACK'S M-64 extends the memory of your ZX81 or Timex-Sinclair 1000 to a full 64-K. It's user transparent. It plugs directly into the back of the ZX81 and has an expansion port to allow you to still use a printer. No extra power supply is required.

All Memories have extra tight contacts in the connector. This combined with a flat-bottom metal case, provide the shielding and stability needed to **completely eliminate memory "crashes"** common with other memory modules.

INSTANT INFORMATION WITH BYTE-BACK'S MD-1 MODEM *only* $119.95 KIT

WIRED AND TESTED $149.95

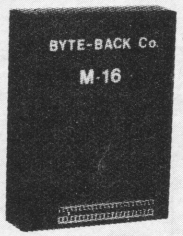
Use your phone to connect your "LITTLE" computer to the "LARGEST" computer networks in the world. With **BYTE-BACK's MD-1 MODEM** connected all you do is dial a phone number (usually local), press a few keys and watch the data appear on your TV screen. (Software is included) This **MODEM** can be used in either the "originate" or "answer" mode with selectable baud rate.

You can have immediate access to:

UNIVERSITY COMPUTERS,DOW JONES, COMPUSERVE & SOURCE

As an extra bonus an RS-232 port is provided to allow you to drive all standard RS-232 peripherals & printers. (75 to 9600 Baud)

EXPAND YOUR 16K SYSTEM (M-16) $59.95 KIT



WIRED and TESTED $69.95

If you have a Sinclair 16K RAM module and need more memory, expand it to 32K and beyond by using **BYTE-BACK M-16 MEMORY MODULES.**

You can't connect two Sinclair 16K RAM modules together, but you can connect one Sinclair 16K and one or more **BYTE-BACK 16K** modules to get all the memory you need.

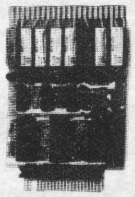
THOUSANDS IN USE WITH PROVEN RELIABILITY

IN STOCK — SAME DAY SHIPMENT

BYTE-BACK'S BB-1 CONTROL MODULE $59.00 KIT *In Stock!*

WIRED and TESTED $69

- **8 Independent Relays** *(with LED status indicators)*
- **8 Independent TTL Inputs** *(with Schmitt trigger buffers)*



- By using a single POKE command you can change and latch the status of each of the 8 relays.
- Your computer can read the status of all 8 inputs by the use of single Peek command.
- A comprehensive manual is included that has complete application details.

Don't settle for a poor quality printer . . .

RS-232 Module $59.95 KIT

WIRED AND TESTED $69.95

Allows you to connect to all RS-232 printers & terminals.

ALL MODULES CARRY 90-DAY WARRANTY

Schematic & Manual included with all bd.s and units.

TRY BYTE-BACK MODULES FOR 10 DAYS WITH NO OBLIGATION

Remember with:* BYTE-BACK *modules you are NOT limited to using only one module at a time!

☐ M-64 Kit	$119.95	☐ M-16 Kit	$59.95
☐ M-64 Wired and Tested	$129.95	☐ M-16 Wired and Tested	$69.95
☐ M-64 Blank PC Board	$19.95	☐ M-16 Blank PC Board	$19.95
☐ BB-1 Kit	$59	☐ Modem Kit	$119.95
☐ BB-1 Wired and Tested	$69	☐ Modem Wired and Tested	$149.95
☐ BB-1 Blank PC Board	$19.95	☐ RS-232 Module Kit	$59.95
		☐ RS-232 Module Wired and Tested	$69.95

Shipping and Handling $4.95

ORDER PHONE (803) 532-5812 **Bill My** ☐ Visa ☐ MasterCard

Exp. Date _____ Card No. _____

Name _____

Address _____

City/State/Zip _____

CHECKS

Mail To: **BYTE-BACK CO.• RT. 3 BOX 147 • BRODIE RD. • LEESVILLE, S.C. 29070**

DEALER DISCOUNT AVAILABLE

BYTE-BACK CO.
RT. 3 BOX 147 • BRODIE RD.
LEESVILLE, S.C. 29070
Ph. (803) 532-5812

BYTE-BACK CO SUPPLIES COMPUTER EQUIPMENT & MODULES TO: **NASA, U.S. GOVERNMENT AGENCIES, BELL LABS, G.E., MARTIN-MARIETA, ALLIED CHEMICAL, & MANY SCHOOLS & UNIVERSITIES.**

```
  10 PRINT "ENTER A NUMBER BETWE
EN 5 AND 9"
  20 INPUT R
  30 IF R<5 THEN PRINT "NUMBER T
OO LOW - REINPUT"
  40 IF R<5 THEN GOTO 10
  50 IF R>9 THEN PRINT "NUMBER T
OO HIGH - REINPUT"
  60 IF R>9 THEN GOTO 10
  70 program continues
```

The second routine checks strings:

```
  10 PRINT "ENTER YES OR NO"
  20 INPUT R$
  30 IF R$<>"YES" AND R$<>"NO" T
HEN PRINT "INCORRECT ENTRY - REI
NPUT"
  40 IF R$<>"YES" AND R$<>"NO" T
HEN GOTO 10
  50 program continues
```

Programmers experienced with using subroutines will note that these input-checking routines would be more efficient if they were written as subroutines. In any event, input-checking routines are important in user-proofing your programs.

Programming Tips

1. Don't use a variable for more than one purpose if you're a beginner. You could easily misuse it and ambush yourself with mystifying errors.

Experienced programmers, however, often recycle variables to save memory. This requires a systematic approach, such as using the variables A, A1, and A$ for temporary storage; R and R$ for user input; and X, Y, and Z for counters or index variables (as in FOR-NEXT loops).

2. List your variables in REM statements at the beginning of your program. This clarifies the program and helps avoid unintentional multiple use of variables.

Include REM statements showing the maximum and minimum limits of numeric variables and the maximum length of string variables, especially if they are to be printed in columns or stored in fixed-length fields.

3. If you use a constant more than once, assign it to a variable. Then you can plug the variable quickly and easily into the program without retyping—and maybe mistyping—the

Translation Table—Variables, LET, INPUT Statements

No "standard" BASIC language exists—only dialects. This table summarizes the main differences between most common microcomputer BASIC dialects and Sinclair BASIC (used in ZX81s and Timex Sinclair 1000s). The table refers only to statements and operations used in or related to the accompanying article. It should help you translate programs written in common dialects into Sinclair BASIC.

Example	What It Does or Is	How to Use It in Common BASIC	Sinclair BASIC	Translation
LET vbl = expr	Assigns value on right to variable on left.	Verb LET is optional. A = 12 is the same as LET A = 12.	LET is required.	Begin statement with LET.
Variable names	Name of memory cell where variable is stored.	Most dialects allow only 1 1 or 2 chars. for a name. 1st must be a letter.	Numeric names any length. String names 1 letter only.	Change name (best to limit numeric to 2 char for portability).
Reserved words (TO, RUN TAN)	All statement, command and function names.	Can't be used anywhere in variable name e.g., TOtal).	No restriction.	Avoid using in variable names (for portability).
Colon(:) (10 LET A = 12 : GOTO 220)	Statement separator. Allows 2 or more statements on one numbered line.	Most dialects allow.	Only 1 statement permitted per line.	Put each statement on separate line
Input prompt (?)	"?" printed on screen as prompt to user.	Most dialects use.	L cursor printed in reverse video on message line.	No programming changes needed—use well written prompt message.
INPUT "message" (INPUT "msg";vbl)	Prints prompt message then waits for user input.	Some dialects allow.	Not allowed.	Put on 2 lines —PRINT "message" on first, INPUT on second.
Multiple inputs (INPUT vbl,vbl)	Allows input to 2 or more variables using 1 INPUT statement.	Many dialects allow (but usually not a good idea to use).	Not allowed.	Use separate INPUT statement for each variable.

Continued Next Page

Example	What It Does or Is	How to Use It in Common BASIC	Sinclair BASIC	Translation
REDO?	Error message when string input is attempted into a numeric variable.	Most dialects use.	Report code 2 appears on message line.	Write input checking routine to prevent error.
LINEINPUT (LINE-INPUT vbl$)	String variable accepts everything input (esp. punctuation marks which cause errors in INPUT statement). Generally used for inputting to sequential access files, sometimes used in sophisticated programs for all input with input checking routines using string functions to analyze the strings.	Some dialects allow.	No LINEINPUT statement, but regular INPUT statement with string variable is similar—accepts all punctuation marks except single quote.	INPUT to string variable.
END	Ends program execution.	Most dialects use, some require as last program line.	Not available.	If END statement is in middle of program, use GOTO to jump to last program line (e.g., line 9999 in Tip 4).

constant. A few examples:

```
LET TAX=.006
LET M$="MESSAGE"
```

4. The INPUT statement provides a handy way to stop a routine and let the user continue at his own speed. Here's a quick example. Assume lines 10, 30 and 50 are lengthy messages that some users will read quickly but others will take time to study before proceeding:

```
 10 PRINT "first message*
 20 GOSUB 1000
 30 PRINT "second message"
 40 GOSUB 1000
 50 PRINT "third message"
 60 GOSUB 1000
 70 ...program continues...
990 GOTO 9999
1000 REM SUBROUTINE TO LET USER
CONTINUE
1010 PRINT "PRESS ENTER TO CONTI
NUE"
1020 INPUT R$
1030 CLS
1040 RETURN
9999 PRINT "END OF PROGRAM"
```

In this example, the program module of lines 1000-1040 is a subroutine that stops the program until the user presses the ENTER key. The input variable must be a string ($) variable; you'll get a syntax error if you try to use a numeric variable, such as R.

Line 990 is the last line of the program before the subroutine. Its function is to jump the program execution to the final line, 9999, which ends the program. Jumps (GOTOs) such as this one are necessary with subroutines to prevent program execution from continuing into the subroutine.

5. Use the same variables in all your programs for temporary responses to INPUT statements (such as R$ in the previous example).

6. If you're going to use a prompt message (such as "PRESS ENTER TO CONTINUE") more than once, assign it early in the program to a string variable (such as M$ for message string). You can then print it when you need it without including the entire message in several PRINT statements.

7. In the FAST mode, the screen will show the display file (data on the screen from prior inputs or computations while prompt is displayed) while the program waits for input data to be entered.

8. INPUT statements won't work in the command mode (entered with no line number). You get a report code 8 if you try. INPUT statements can be used only in numbered program lines.

9. If the user trys to stop your program's execution by entering a STOP statement in response to a request for numeric input, he or she will get a report code D. This is not true if the request is for string (alphanumeric) input. ZX81 manuals don't indicate this, so make a note.

A Program To Index Articles— 8K ROM, 16K RAM

This program lets you index and instantly find articles from your favorite publications five ways: by name, category, volume, issue and page. *Articles Index* as listed is set up for SYNTAX newsletter.

To use the program for the first time, simple type it in and enter RUN. From there, choose the APPEND routine and enter the information the program requests for as many articles you want to put in at that time. In an attempt to save memory, the only way to exit the APPEND routine is to enter shift 5 and then shift A. At this point you may wish to enter a new line 10 telling you to utilize the program by entering GOTO 80.

To list the programs and articles, enter GOTO 80 and push 2 to take you to the LIST routine. A menu appears from which you can choose how you want to select the articles. To list all the articles or programs, enter 1. To list articles and programs by name, enter 2. From here you can enter any unique part of the article name, and it will list all with that fragment. To list by category, enter the first two letters of the category. To list by year, simply enter either the year or the volume number. For all but the first volume of SYNTAX, the issue number and the number of the month are the same, so when listing by month or issue (number 5 in the choices), enter that number. The computer asks for the year or volume here. If you want a specific year, enter that year; if not enter 0 and you will get all years. By page, enter the year, month, and page (in that order). Once again, if you care not to be specific about either year (vol), or month (issue), then enter 0 when the computer requests these.

The program flashes the list on the screen fairly quickly. If you wish to view the list for a longer time, press P when it flashes on the screen, and the computer will hold that screen until you press any key.

Add more articles at any time by going to the APPEND routine and using it as before.

After your first session, save *Articles Index* on tape. All the listings you entered will be saved with the program. The next time you want to use *Articles Index*, start the program with GOTO 80. Never use RUN. RUN will erase all the listings you saved.

Brief Description of Subroutine Functions:

10-160	Dimension variables to use, print introduction, and go to the proper routine.
170-240	Send you to the proper subroutine to list articles as you requested.
250-290	Print the information on the screen from the information you requested, and let you leave the same screen for a while to read it.
300-330	Print all articles.
350-395	Print articles found with the name you gave.
400-445	Print all articles in a certain category.
450-495	Print all articles from a certain year or volume.
500-545	Print all articles in a specific issue or all articles from the same issues from different years.
550-610	Print all articles on one page from one issue, a year's worth of issues, or all issues.
1000-1210	Find the place to start inserting information and put information you enter into the data files.

Banta Software also offers other programs for practical use or for fun and games. If you are looking for any program in particular, please write—I specialize in customized programs for all purposes. **SQ**

Andrew Q. Banta
RD #7
Bethelem, PA 18015

```
 10 DIM A$(800)
 20 DIM B$(400,24)
 30 DIM C$(400,5)
 80 PRINT AT 8,8;"SYNTAX INDEX"
120 PRINT AT 9,10;"1. APPEND"
130 PRINT AT 10,10,;"2. LIST"
140 PAUSE 50000
150 POKE 16437,255
160 IF INKEY$="1" OR INKEY$="A"
THEN GOTO 1000
```

Continued Next Page

```
  170 CLS
  180 PRINT AT 8,8;"1. ALL"
  190 PRINT AT 9,8;"2. BY NAME"
  192 PRINT AT 10,8;"3. BY CATEGO
RY"
  195 PRINT AT 11,8;"4. BY VOL/YE
AR"
  200 PRINT AT 12,8;"5. BY MONTH/
ISSUE"
  205 PRINT AT 13,8;"6. BY PAGE"
  220 INPUT K
  230 IF K<1 OR K>6 THEN GOTO 220
  235 CLS
  240 GOTO K*50+250
  250 SCROLL
  260 PRINT " ";C$(I);"-";B$(I)
  265 SCROLL
  270 PRINT "#VOL/ISSUE/PAGE  ";A
  "/";B;"/";C
  272 SCROLL
  275 PAUSE 90
  280 IF INKEY$="P" THEN PAUSE 50
000
  285 POKE 16437,255
  290 RETURN
  300 FOR I=1 TO 400
  305 LET A=INT ((CODE A$(I)-1)/1
2)
  310 LET B=CODE A$(I)-A*12
  315 LET C=CODE A$(I+400)
  320 GOSUB 250
  325 NEXT I
  330 STOP
  350 PRINT "ENTER ANY PART OF NA
ME"
  355 INPUT Z$
  360 FOR I=1 TO 400
  362 FOR J=1 TO 24-LEN Z$
  365 IF Z$=B$(I,J TO J+LEN Z$-1)
  THEN GOTO 370
  366 NEXT J
  367 GOTO 390
  370 LET A=INT ((CODE A$(I)-1)/1
2)
  375 LET B=CODE A$(I)-A*12
  380 LET C=CODE A$(I+400)
  385 GOSUB 250
  390 NEXT I
  395 STOP
  400 PRINT "ENTER THE FIRST 2 LE
TTERS OF     CATEGORY"
  405 INPUT Z$
  410 FOR I=1 TO 400
  415 IF Z$(1 TO 2)<>C$(I,1 TO 2)
  THEN GOTO 440
  420 LET A=INT ((CODE A$(I)-1)/1
2)
  425 LET B=CODE A$(I)-A*12
  430 LET C=CODE A$(I+400)
  435 GOSUB 250
  440 NEXT I
  445 STOP
  450 PRINT "ENTER VOLUME OR LAST
  2 DIGITS OFYEAR"
  455 INPUT Z
  460 IF Z>79 THEN LET Z=Z-79
  462 FOR I=1 TO 400
  465 LET A=INT ((CODE A$(I)-1)/1
2)
  470 LET B=CODE A$(I)-A*12
  475 LET C=CODE A$(I+400)
  480 IF Z<>A THEN GOTO 490
  485 GOSUB 250
  490 NEXT I
  495 STOP
  500 PRINT "ENTER VOL OR LAST 2
DIGITS OF    YEAR, MONTH OR ISSUE
. ENTER 0    FOR YEAR IF YOU WANT
 ALL OF      ISSUE FROM ALL YEARS
 (HIT N/L    AFTER BOTH)."
  501 INPUT Y
  502 INPUT Z
  503 IF Y>79 THEN LET Y=Y-79
  505 FOR I=1 TO 400
  510 LET A=INT ((CODE A$(I)-1)/1
2)
  515 LET B=CODE A$(I)-A*12
  520 LET C=CODE A$(I+400)
  525 IF (Z=B AND Y=A) OR (Z=B AN
D Y=0) THEN GOTO 535
  530 GOTO 540
  535 GOSUB 250
  540 NEXT I
  545 STOP
  550 PRINT "ENTER VOL OR YEAR, I
SSUE OR      MONTH, PAGE. ENTER 0
 FOR ANY     CATEGORY (EXCEPT PAG
E) YOU WISH TO NOT BE SPECIFIC A
BOUT(HIT     N/L AFTER EACH)."
  551 INPUT Y
  552 INPUT Z
  553 INPUT X
  554 IF Y>79 THEN LET Y=Y-79
  555 FOR I=1 TO 400
  560 LET A=INT ((CODE A$(I)-1)/1
2)
  565 LET B=CODE A$(I)-A*12
  570 LET C=CODE A$(I+400)
  580 IF (X=C AND Y=A AND Z=B) OR
  (X=C AND Y=0 AND Z=B) OR (X=C A
ND Y=0 AND Z=0) THEN GOTO 590
  585 GOTO 600
  590 GOSUB 250
  600 NEXT I
  610 STOP
 1000 FOR I=1 TO 400
 1005 CLS
 1010 IF C$(I,1)=" " THEN GOTO 10
50
 1020 NEXT I
 1030 PRINT "NO ROOM LEFT"
 1040 STOP
 1050 FOR I=1 TO 400
 1060 PRINT "ENTER CATEGORY"
 1070 INPUT C$(I)
 1080 PRINT "ENTER NAME"
 1090 INPUT B$(I)
 1100 PRINT "ENTER VOLUME OR YEAR
"
 1110 INPUT Z
 1115 IF Z>79 THEN LET Z=Z-79
 1120 PRINT "ENTER MONTH"
 1130 INPUT Y
 1140 PRINT "ENTER PAGE"
 1150 INPUT X
 1160 LET A$(I+400)=CHR$ X
 1170 LET X1=Z*12+Y
 1180 LET A$(I)=CHR$ X1
 1190 CLS
 1200 NEXT I
 1210 PRINT "NO ROOM LEFT"
SYNTACTIC SUM:  27041, 8K ROM
```

Improved Super Monzxer Game

Super Monzxer originally appeared in SYNTAX newsletter in Nov. 81. This listing includes changes and improvements to Banta Software's Super Monzxer program.

You are trapped in a multi-level cave with many rooms. Monzxers lurk, ready to eat you. Your only defense is a sword hidden in one of the rooms. To emerge from the cave alive, you must kill the monzxers.

Along the way, other hazards try to kill you. Man-eating spiders, giant bats and bottomless pits can end your life (and the game) prematurely.

If you enter a spider room, you must shoot it or it will kill you. To shoot, select S when the computer asks, "MOVE OR SHOOT?" Then select the five rooms of your arrow's flight. But be careful: if the arrow's path brings it back to your room, you shoot yourself.

Pits drop you unexpectedly into another level and room. Great for escape—but the room you land in may not be safe!

Bats can pick you up and move you randomly, again to a potentially dangerous room.

If you enter a Monzxer room, you are eaten instantly unless you have a sword. When you enter the sword room, it is yours to keep.

But your friendly ZX/TS computer warns you of approaching danger. "I FEEL A DRAFT" tells you that you are near a pit. As you approach a spider room, your computer prints, "I SEE COBWEBS." "BATS NEARBY" warns of giant bats. Most threateningly, "I SMELL A MONZXER" tells of a nearby Monzxer.

The computer also tells you your present location in the cave after each move by level and room number. It also tells you which rooms on the same level you can move to. You choose whether to move or shoot. If you move, enter the number of the room to go to.

To use Super Monzxer, just type in the listing as shown and RUN. Here's a key to the graphics lines. GS means press SHIFT 9 to get a G graphics cursor and hold the SHIFT key while pressing the key following. For example, GS6 means press SHIFT 9 then SHIFT 6 to get the graphics symbol. G means press SHIFT 9 to get the G graphics cursor, then press the key following. For example, GQ means press SHIFT 9 then Q to get an inverse Q.

Line	Enter
1423	/ GS6 GSF GSF GSF GS6 / 3 spaces Gspace Gspace
1424	Gspace GQ GS7 GS7 GS7 GQ Gspace 2 spaces Gspace Gspace Gspace Gspace Gspace Gspace
1425	Gspace Gspace GSW GS6 GSQ Gspace Gspace GSH Gspace Gspace Gspace Gspace Gspace Gspace Gspace Gspace /
3510	/ Gspace Gspace Gspace Gspace Gspace Gspace Gspace Gspace GSH Gspace Gspace GSE GS7 GSR Gspace Gspace
3520	> > = = = GSL GSL GSM Gspace Gspace Gspace Gspace space Gspace GQ GS6 GS6 GS6 GQ Gspace
3530	Gspace Gspace Gspace Gspace Gspace 2 spaces / GSG GSG GSG GS7 /
3540	Gspace Gspace Gspace

To exit early, press BREAK when the computer asks, "MOVE OR SHOOT?"

If you prefer not to wait while the computer dimensions arrays, delete lines 398 and 1005. **SQ**

```
    1 REM "MONZXER"
    2 RAND
    5 LET END=0
   10 PRINT AT 10,4;"AN A. G. BAN
TA PRODUCTION"
   15 PRINT AT 12,0;"EDITED TO MA
KE MONZXER WORK     BETTER BY R.
 J.  CALHOON","2618 ROSELAWN AVE
.","BALTO., MD 21214"
   20 PAUSE 500
   30 CLS
   50 PRINT AT 10,20;"SUPER"
   60 PRINT AT 12,18;"MONZXER"
   70 FOR I=3 TO 37
   80 FOR J=11 TO 19
   87 UNPLOT 9,16
   90 PLOT J,I
  100 NEXT J
  110 NEXT I
  120 FOR I=16 TO 24
  130 FOR J=8 TO 22
```

Continued Next Page

```
140 PLOT J,I
150 NEXT J
160 NEXT I
170 FOR I=22 TO 32
180 PLOT 6,I
190 PLOT 24,I
200 NEXT I
210 PLOT 5,22
220 PLOT 7,22
225 PLOT 7,21
230 PLOT 23,22
235 PLOT 25,22
240 PLOT 23,21
245 PLOT 25,21
250 PLOT 9,35
255 PLOT 10,35
265 PLOT 10,36
270 PLOT 20,36
275 PLOT 20,35
280 PLOT 21,35
285 FOR I=14 TO 16
290 FOR J=3 TO 16
295 UNPLOT I,J
300 NEXT J
305 NEXT I
307 PLOT 23,33
312 PLOT 7,33
315 PLOT 7,32
320 FOR I=5 TO 25
325 IF I=14 THEN LET I=17
330 PLOT I,3
335 NEXT I
338 FOR I=6 TO 24
340 IF I=14 THEN LET I=17
342 PLOT I,4
345 NEXT I
350 PRINT AT 21,3;"""   """
352 PRINT AT 21,9;"""   """
355 PRINT AT 7,5;"0"
357 PRINT AT 7,9;"0"
360 PRINT AT 9,5;"      "
365 PRINT AT 5,22;" ----/"
370 PRINT AT 4,28;"/"
375 PRINT AT 3,29;"/"
380 PRINT AT 2,30;"/"
382 UNPLOT 8,17
385 UNPLOT 8,16
387 UNPLOT 9,16
390 UNPLOT 21,16
392 UNPLOT 22,16
393 UNPLOT 22,17
395 SLOW
396 IF END=1 THEN STOP
397 PAUSE 600
398 FAST
399 DIM R(3,101)
400 DIM A(6)
401 FOR I=1 TO 3
402 FOR J=1 TO 101
403 LET R(I,J)=INT (RND*100)+1
404 NEXT J
405 NEXT I
406 LET SW=0
407 LET DM=0
408 FOR L=0 TO 80 STEP 20
410 FOR I=1 TO 20
415 LET K=L+I
420 FOR J=1 TO 3
422 LET J1=J
425 IF J1<INT ((I-1)/5) THEN LE
T J1=J1-1
430 IF R(J,K)>0 THEN GOTO 495
435 LET G=INT (RND*5)+J1*5+L+1
436 FOR M=1 TO 5
437 IF R(J,L+M+INT ((I-1)/5)*5)
=G THEN GOTO 435
438 NEXT M
440 FOR H=1 TO 3
445 IF R(H,G)=K THEN GOTO 435
450 IF R(H,G)=0 THEN GOTO 465
455 NEXT H
460 GOTO 435
465 LET R(J,K)=G
490 LET R(H,G)=K
495 NEXT J
500 NEXT I
510 NEXT L
525 LET YR=INT (RND*20)+1
535 LET S1=5
610 DIM X(36)
620 DIM T(30)
630 DIM C(30)
640 DIM Q(30)
650 DIM W(3)
660 DIM S(10)
670 DIM B(10)
680 DIM P(10)
690 LET DM=0
695 LET SW=0
700 LET SR=INT (RND*100)+1
710 FOR I=1 TO 3
720 LET W(I)=INT (RND*100)+1
725 IF W(I)=YR THEN GOTO 720
730 NEXT I
740 FOR I=1 TO 9 STEP 2
745 FOR J=0 TO 1
750 LET S(I+J)=(INT (RND*20)+1)
+20*INT (I/2)
760 LET B(I+J)=(INT (RND*20)+1)
+20*INT (I/2)
770 LET P(I+J)=(INT (RND*20)+1)
+20*INT (I/2)
780 NEXT J
790 NEXT I
800 FOR I=0 TO 9
810 FOR J=1 TO 3
820 LET T(I*3+J)=R(J,S(I+1))
830 LET C(I*3+J)=R(J,B(I+1))
840 LET Q(I*3+J)=R(J,P(I+1))
850 NEXT J
860 NEXT I
870 FOR I=0 TO 2
880 FOR J=1 TO 3
890 LET X(I*3+J)=R(J,W(I+1))
900 FOR L=1 TO 3
910 LET GG=((J-1)*3)+(I*9)+L+9
920 LET X(GG)=R(L,X(I*3+J))
930 NEXT L
950 NEXT J
960 NEXT I
1000 CLS
1005 SLOW
1010 LET YL=INT ((YR-1)/20)+1
1015 IF YL>5 THEN LET YL=1
1020 PRINT "YOU ARE ON LEVEL ";Y
L
1024 LET YR=YR-(YL-1)*20
1026 IF YR>99 THEN LET YR=INT (R
ND*100)-1
1030 PRINT "YOU ARE IN ROOM ";YR
1032 IF YR=SR THEN PRINT "THERE
IS A SWORD IN THE ROOM"
1037 IF YR=SR THEN LET SW=1
1040 FOR I=1 TO 30
1050 IF YR=T(I) THEN GOTO 1080
1060 NEXT I
1070 GOTO 1090
1080 PRINT "I SEE COBWEBS"
1090 FOR I=1 TO 36
1100 IF YR=X(I) THEN GOTO 1130
1110 NEXT I
1120 GOTO 1140
1130 PRINT "I SMELL A MONZXER"
1140 FOR I=1 TO 30
1150 IF YR=Q(I) THEN GOTO 1180
1160 NEXT I
1170 GOTO 1190
1180 PRINT "I FEEL A DRAFT"
1190 FOR I=1 TO 30
1200 IF YR=C(I) THEN GOTO 1230
1210 NEXT I
1220 GOTO 1240
1230 PRINT "BATS NEARBY"
1240 FOR I=1 TO 3
1250 IF YR=W(I) THEN GOTO 1280
1260 NEXT I
1270 GOTO 1300
1280 PRINT "THERE IS A MONZXER I
N YOUR ROOM"
1281 IF SW=1 THEN GOTO 1285
1282 PRINT "AND IT ATE YOU,DUMMY
"
1284 GOTO 3000
1285 PRINT "BUT YOU KILLED IT WI
TH YOUR SWORD"
1290 PAUSE 300
1292 LET DM=DM+1
1294 IF DM=3 THEN GOTO 2990
1295 IF I>3 THEN LET I=1
```

Continued Next Page

```
1297 LET W(I)=101
1298 GOTO 870
1300 FOR I=1 TO 10
1310 IF YR=B(I) THEN GOTO 1340
1320 NEXT I
1330 GOTO 1380
1340 CLS
1350 LET YR=(INT (RND*20)+1)+(YL
-1)*20
1360 PRINT "BATS TOOK YOU TO ROO
M ";YR-(YL-1)*20
1370 GOTO 1010
1380 FOR I=1 TO 10
1390 IF YR=S(I) THEN GOTO 1420
1400 NEXT I
1410 GOTO 1430
1420 PRINT "THERE IS A SPIDER IN
 YOUR ROOM  AND HE IS HUNGRY"
1423 PRINT AT 9,10;"/▓▓▓▓▓▓▓/   ▓▓
"
1424 PRINT AT 10,9;"▓▓ ▓▓  ▓▓  ▓▓
▓▓▓▓"
1425 PRINT AT 11,9;"▓▓▓▓▓▓▓▓▓▓▓▓▓
▓▓▓▓▓/"
1427 PRINT AT 12,17;"IL IL IL"
1428 PRINT AT 14,7;"YUM YUM EATE
M UP"
1429 PAUSE 100
1430 FOR I=1 TO 10
1440 IF YR=P(I) THEN GOTO 1470
1450 NEXT I
1460 GOTO 1540
1470 IF YL=5 THEN GOTO 1520
1480 CLS
1490 PRINT "TSK, TSK, YOU FELL I
N A PIT"
1500 LET YR=YR+20
1505 IF YL>=2 THEN GOTO 1540
1510 GOTO 1010
1520 PRINT "YOU FELL INTO A BOTT
OMLESS PIT  YOU TURKEY"
1530 GOTO 3000
1540 PRINT "TUNNELS TO ROOMS ";
1550 FOR I=1 TO 3
1560 PRINT " ";ABS (R(I,YR)-(YL-
1)*20);
1570 NEXT I
1580 PRINT
1590 PRINT "YOU MUST MOVE OR SHO
OT"
1600 PAUSE 900
1610 LET K$=INKEY$
1620 IF K$="M" THEN GOTO 1650
1630 IF K$="S" THEN GOTO 2000
1640 GOTO 1600
1650 FOR I=1 TO 10
1652 IF YR=S(I) THEN GOTO 1657
1654 NEXT I
1656 GOTO 1659
1657 PRINT "THE SPIDER IS MUNCHI
NG ON YOU"
1658 GOTO 3000
1659 PRINT "WHICH ROOM?"
1660 INPUT YR1
1670 FOR I=1 TO 3
1680 IF YR1=ABS (R(I,YR)-(YL-1)*
20) THEN GOTO 1720
1690 NEXT I
1700 PRINT "NO TUNNEL TO THIS RO
OM...SIGNED, MONXZER"
1710 GOTO 1650
1720 LET YR=YR1+(YL-1)*20
1730 GOTO 1000
2000 LET S1=S1-1
2005 FOR I=1 TO 10
2010 IF YR=S(I) THEN GOTO 2035
2011 IF K$="S" THEN PRINT "MONZX
ER=Z OR SPIDER=D"
2012 PAUSE 900
2014 LET S$=INKEY$
2015 IF S$="Z" THEN GOTO 2060
2016 IF S$="D" THEN GOTO 2060
2017 GOTO 2012
2020 NEXT I
2030 GOTO 2060
2035 CLS
2036 GOSUB 3500
2037 PRINT AT 18,5;"YOU KILLED T
HE SPIDER..."
2038 PRINT "▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓"
2040 PAUSE 300
2043 IF I>10 THEN LET I=1
2045 LET S(I)=101
2050 GOTO 800
2060 CLS
2070 PRINT "MAKE A LIST OF 5 ROO
MS FOR THE  ARROWS FLIGHT"
2080 FOR I=2 TO 6
2090 INPUT A(I)
2100 PRINT A(I)
2105 LET A(I)=A(I)+(YL-1)*20
2110 NEXT I
2120 LET A(1)=YR
2130 FOR I=2 TO 6
2140 FOR J=1 TO 3
2150 IF A(I)=R(J,A(I-1)) THEN GO
TO 2180
2160 NEXT J
2165 LET J=INT (RND*3)+1
2170 LET A(I)=R(J,A(I-1))
2180 FOR J=1 TO 3
2190 IF A(I)=W(J) AND S$="Z" THE
N GOTO 2220
2195 IF YR=S(I) AND S$="D" THEN
GOTO 2215
2200 NEXT J
2210 GOTO 2230
2215 PRINT "YOU SLEW A SPIDER"
2217 GOSUB 3500
2218 GOTO 2040
2220 PRINT "YOU SLEW A MONZXER"
2221 IF J>3 THEN LET J=1
2222 LET W(J)=101
2223 PAUSE 300
2224 LET DM=DM+1
2225 IF DM=3 THEN GOTO 2990
2225 IF S1=0 THEN GOTO 2283
2228 GOTO 870
2230 IF NOT A(I)=YR THEN GOTO 22
60
2240 PRINT "YOU SHOT YOURSELF"
2250 GOTO 3000
2260 NEXT I
2265 IF S$="D" THEN GOTO 2275
2270 PRINT "YOU MISSED THE MONZX
ER"
2273 GOTO 2280
2275 PRINT "YOU MISSED THE SPIDE
R"
2280 PAUSE 300
2283 IF S1=0 THEN PRINT "YOU ARE
 OUT OF ARROWS"
2285 IF S1=0 THEN GOTO 3000
2290 GOTO 1000
2990 PRINT "NO MORE MONZXERS YOU
 WIN ...SHUCKS"
3000 PAUSE 600
3005 CLS
3010 PRINT AT 10,10;"PLAY AGAIN?
"
3020 PAUSE 900
3030 LET K$=INKEY$
3040 IF K$="Y" THEN GOTO 3070
3050 IF K$="N" THEN GOTO 3130
3060 GOTO 3020
3070 PRINT AT 10,6;"SAME TUNNEL
SET?"
3080 PAUSE 900
3090 LET K$=INKEY$
3100 IF K$="Y" THEN GOTO 525
3110 IF K$="N" THEN GOTO 399
3120 GOTO 3030
3130 CLS
3140 PRINT AT 10,20;"HAR HAR"
3150 PRINT AT 12,18;"YOU LOSE"
3160 LET END=1
3165 FAST
3170 GOTO 70
3500 PRINT AT 10,9;"TT TT TT"
3510 PRINT AT 11,8;"/▓▓▓▓▓▓▓▓▓▓▓▓
▓▓▓▓▓▓▓▓"
3520 PRINT AT 12,5;">>===▓▓▓▓▓▓▓▓
 ▓▓  ▓▓"
3530 PRINT AT 13,11;"▓▓▓▓▓▓  /▓▓▓▓
▓/"
3540 PRINT AT 14,12;"▓▓▓▓"
3542 PRINT AT 9,20;"?  ?"
3544 PRINT AT 7,24;"?"
3546 PRINT AT 5,27;"?"
3550 PRINT AT 16,22;"YA GOT ME"
3560 RETURN
SYNTACTIC SUM:  55885, 8K ROM
```

NEW! For the first time— *AVAILABLE IN THE USA!*

4 NEW Handbooks for your Sinclair ZX-81 or Timex-Sinclair 1000!!

MASTERING MACHINE CODE ON YOUR ZX-81

by Toni Baker

Until this comprehensive, yet easy-to-understand, handbook, there was virtually no material available about the ZX machine code. Using this guide you'll learn the ins and outs of ZX machine code translation. Discover the secrets of the ZX-81, and even see how to adapt the code to the ZX-80 machine. When you understand the language translations between BASIC and the ZX machine code, you'll enjoy the workings of your computer to the utmost!

49 EXPLOSIVE GAMES FOR THE ZX-81

Edited by Tim Hartnell

Galactic Intruders, Breakout, Checkers, Death Maze, Smugglers Mold, and forty-three other favorites are all here, newly adapted especially for you and your new ZX-81 personal computer. This fascinating gamebook gives you programming instructions for all 49 marvelous games PLUS complete and easy-to-understand game rules. This wonderfully exciting hardcover playbook can be yours, order below.

MAKING THE MOST OF YOUR ZX-81

by Tim Hartnell

All new for you and your new ZX-81, this handbook focuses on the additional features of the ZX-81. You'll have new games and useful learning tricks, and you'll also see how to write programs that really work. It will guide you through start to finish, using each feature and function of your new ZX-81 personal home computer.

THE ZX-81 POCKETBOOK

by Trevor Toms

This handy new programming manual really gets you into ZX-81 functions. Don't just type someone else's programs . . . now you can create your *own* and understand why they work. It's fun to learn all about computing with the ZX-81 POCKET BOOK as a guide. You'll see what your new ZX-81 can do, and what extras will make it able to do even more . . . see how to use ZX-81 BASIC in the best ways . . . learn to avoid frustration and retyping with program and data file storage and retrieval techniques—*and* for ZX-80 owners, you'll learn how to convert your ZX-80 to the advanced ZX-81 capabilities. And there's so much more! This road map to the ZX-81 can be yours—it's worth every penny!

Using these books you'll be amazed at how fast you'll become a ZX-81 wizard. Of course, if you don't find the books helpful and interesting return them within 15 days for a full refund, and owe nothing.

RESTON PUBLISHING COMPANY c/o
Mail to: PRENTICE-HALL INC., Book Distribution Center
Route 59 at Brook Hill Drive, West Nyack, New York 10995

Please send me my 15-day trial copies of the following titles:

____ **MASTERING MACHINE CODE ON YOUR ZX-81,** by Toni Baker, (R4262-3), $18.95
____ **49 EXPLOSIVE GAMES FOR THE ZX-81,** by Tim Hartnell, (R2087-6), $16.95
____ **MAKING THE MOST OF YOUR ZX-81,** by Tim Hartnell, (R4189-8), $16.95
____ **THE ZX-81 POCKET BOOK,** by Trevor Toms, (R9525-8), $16.95

Name_____
Address_____
City_____ State_____ Zip_____

VISA

Now you can charge your orders! Just fill in the information below:
VISA ☐ ☐ MASTERCHARGE

master charge

account #_____ expiration date_____
signature_____

Save! If check or money order, plus your state's sales tax, accompanies your order, publisher pays all postage and handling charges. Same money back guarantee applies.

Dept. V V-0885-3A(9)

Syntactic Sum with Variables

When loading a program into your ZX/TS from tape, how do you know the program loaded properly? If even one bit was mishandled in the transfer from tape to computer, the program could run improperly or not at all.

One way to check is to run a Syntactic Sum immediately after loading from tape, and compare the results to the known good Syntactic Sum. However, the Syntactic Sum described in the Jan. 82 issue of SYNTAX newsletter checks only the program area of RAM, from 16509 to D_FILE. It does not check the variables section of RAM. Many programs require that variables be saved onto tape along with program, such as checkbook or data file programs. Therefore, a Syntactic Sum that also checks the variables section of RAM is desirable for checking the integrity of a program with stored data after loading from tape.

Syntactic Sum with Variables is a revised version of SYNTAX's Syntactic Sum. This version checks the variables section of RAM. It is compatible with Syntactic Sums listed after programs in SYNTAX newsletter and SQ, as long as you do the sum immediately after typing in the program and before running it (running the program sets up variable storage). The machine language program is placed above RAMTOP for safety even from NEW. Once it is loaded at the beginning of a computing session, Syntactic Sum is always there until you turn the computer off. Syntactic Sum with Variables is written for machines with 8K ROM and 16K RAM only. If you have a working knowledge of Z80 machine code, you could rewrite it for different memory sizes.

How to Make and Use a Syntactic Sum with Variables Tape for 8K ROM, 16K RAM ZX/TS Computers

First lower RAMTOP by following these steps:

```
POKE 16388,202
POKE 16389,127
NEW
```

Now create the Syntactic Sum with Variables tape. Key in the following:

```
1 DIM P(50)
2 FOR L=1 TO 50
3 PRINT "INPUT P(";L;")";
4 INPUT P(L)
5 PRINT P(L),
6 NEXT L
7 STOP
```

RUN the program and key in the numbers from this decimal listing. Read left to right and top to bottom:

```
 33 125  64 237  91  12  64 253  33   0   0
205 238 127  42  16  64 237  91  20  64  27
205 238 127 253 229 193 253  33   0  64 201
124 186  32   5 125 187  32   1 201  78   6
```

The screen will now be full. Check your entries for accuracy, then key in CONT and ENTER. Now key in the remaining numbers:

```
 0 253   9  35  24 239
```

From this point on, *DO NOT* enter RUN or you will erase the array you just created. Enter LIST and key in the following lines:

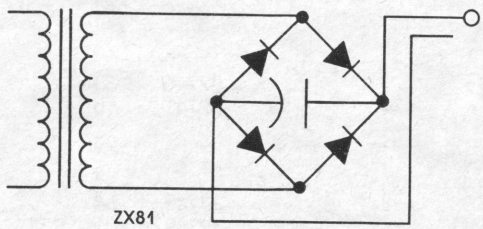
```
   1 FOR L=32717 TO 32766
   2 POKE L,P(L-32716)
   3 NEXT L
   4 PRINT "SYNTACTIC SUM WITH V
ARIABLES IS NOW SAFELY TUCKED AW
AY ABOVE RAM TOP"
   5 PRINT '"TO OBTAIN A SYNTAC
TIC SUM, MAKE SURE THE COMPUTER
IS IN FAST MODE"
   6 PRINT '"THEN ENTER ""PRINT
 USR 32717""
   7 STOP
   8 SAVE "SUM"
   9 GOTO 1
```

Now start your tape recorder. Save the program to tape by entering GOTO 8. Your Syntactic Sum with Variables tape is done. (If you enter PRINT USR 32717 now, you'll get 24778. If you list this program, the M in line 8 will be in reverse video. This is normal for self-starting programs after execution.—Ed.)

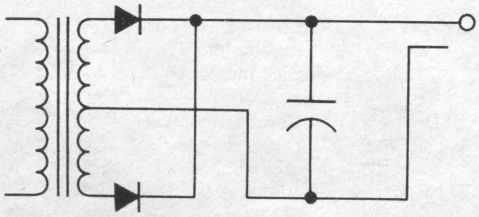
Continued on Page 18

ZX/TS Power Supplies

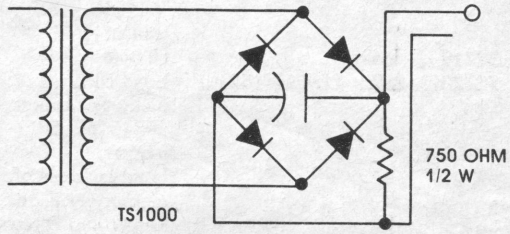
Property	ZX80	ZX81	TS1000
Input Power (Watts)	8	12	15
Rated Volts	9	9.75	9
Rated Current (mA)	500	650	1000
Voltage (DC)			
Open-Circut	16.3	14.6	13.1
Loaded by ZX81	10.4	11.3	11.2
Add 16K RAM	7.9	10.2	10.3
Ripple (Volts p-p)			
Open-Circuit	<0.005	<0.005	0.075
Loaded by ZX81	1.7	2.0	1.0
Add 16K RAM	2.3	3.0	1.5
Filter Capacitor			
Size—MicroF	1000	1000	2200
Rating—Volts	16	16	25
Transformer			
Resistance (Ohms)			
Primary	260	60	30
Secondary	2.2 per side	0.6	0.6
E-Height (in.)	1.125	1.0625	1.25
Stack Height (in.)	0.5	1.0	0.78
Construction			
Laminations	Butted & Clipped	Butted & Clipped	Butted & Taped
Bobbin	Layered	Layered	Side-by-side
Circuit	Full-wave Center-tapped	Full-wave Bridge	Full-wave Bridge
Total Weight (gm.)	222.6	351.3	369.8



ZX81



ZX80



TS1000

To use your new tape, follow these steps after power-up (plugging in your computer):

POKE 16388,202	(these two POKEs lower RAMTOP)
POKE 16389,127	
NEW	(relocates stack)
LOAD "SUM"	(from tape; it automatically runs itself with instructions)

To conveniently use Syntactic Sum with Variables to check programs after loading from tape, follow this procedure: Put Syntactic Sum with Variables above RAMTOP as described earlier (POKE 16388,202 POKE 16389,127 NEW). Then save your programs onto tape in the following format, where XXX, YYY and ZZZ are line numbers at the end of your program (line 5 is not needed unless you have an 8K ROM produced before September, 1981, containing a bug. Newer ROMs have no bug):

```
  1 PRINT "SYNTACTIC SUM WITH
VARIABLES = "
  2 PRINT USR 32717
  3 PRINT ,,"PRESS ANY KEY TO C
ONTINUE"
  4 PAUSE 32800
  5 POKE 16437,255
(your program follows)
XXX STOP
YYY SAVE "NAME OF YOUR PROGRAM"
ZZZ GOTO 1
```

Make sure your computer is in the FAST mode. Turn on your recorder, then save to tape by entering GOTO YYY. After the program is saved, the computer will automatically run the first four lines, displaying a Syntactic Sum with Variables. This sum is for the program exactly as saved. Copy this number on the tape label. Any time you load this tape, you should see this exact number on your screen. If the number isn't identical, the program didn't load properly. Try reloading.

You can also check existing tapes by entering PRINT USR 32717 after loading, but I recommend modifying them to this "built-in" method for convenience. **SQ**

Donald James Beck is a member of ZIPPER (ZX81 Interactive Programs and Peripherals Exchange Resource) Users' Group.

Syntactic Sum with Variables Assembly Listing

Address	Code	Mnemonic	Comment
32717	33	LD HL,16509	;Load HL register ;with the 2-byte ;integer 16509
32718	125		;lo byte— ;(64*256) + 125 = 16509 ;start of RAM
32719	64		;Hi byte
32720	237	LD DE,(16346)	;Load contents of ;memory location ;16346 into E ;register
32721	91		;Load contents of ;memory location ;16347 into D register
32722	12		;Lo byte— ;(64*256) + 12 = 16346 D_FILE
32723	64		;Hi byte— ;16396 + 1 = 16347 D_FILE
32724	253	LD IY,0000	;Load IY register ;with the 2-byte ;integer 0000
32725	33		
32726	0		;Starts summation at ;zero
32727	0		
32728	205	CALL 32750	;Call subroutine ;located at address ;32750
32729	238		;Lo byte— ;(127*256) + 238 = 32750
32730	127		;Hi byte
32731	42	LD HL,(16400)	;Load contents of ;memory location NN ;into L register and ;NN + 1 into H ;register
32732	16		;Lo byte— ;(64*256) + 16 = 16400 VARS
32733	64		;Hi byte— ;16400 + 1 = 16401 VARS
32734	237	LD DE,(16404)	;Load contents of ;memory location ;16404 into E ;register
32735	91		;Load contents of ;memory location ;16405 into D register
32736	20		;Lo byte— ;(64*256) + 20 = 16404 ;E_LINE
32737	64		;Hi byte— ;16404 + 1 = 16405 ;E_LINE
32738	27	DEC DE	;Decrement DE ;register
32739	205	CALL 32750	;Call subroutine ;located at address ;32750
32740	238		;Lo byte— ;(127*256) + 238 = 32750
32741	127		;Hi byte
32742	253	PUSH IY	;Push contents of IY ;register onto ;stack
32743	259		
32744	193	POP BC	;Pop contents of ;stack into BC ;register
32745	253	LD IY,16384	;Load IY register ;with the 2-byte ;integer 16384
32746	33		
32747	0		;Lo byte—loads IY ;register with 4000h ;as per chap. 26 ZX81 ;manual
32748	64		;Hi byte
32749	201	RET	;Return to BASIC
32750	124	LD A,H	;Load contents of H ;register into ;accumulator (A ;register)
32751	186	CP D	;Compare contents of ;D register with ;accumulator
32752	32	JR NZ,32759	;Jump relative if Z ;flag is zero
32753	5		;Displacement factor

Address	Code	Mnemonic	Comment
32754	125	LD A,L	;Load contents of L ;register into ;accumulator
32755	187	CP E	;Compare contents of ;E register with ;accumulator
32756	32	JR NZ,32759	;Jump relative if Z ;flag is zero
32757	1		;Displacement factor
32758	201	RET	;Return from ;subroutine
32759	78	LD C,(HL)	;Load contents of ;memory location ;pointed to by HL ;register into C ;register
32760	6	LD B,0	;Load following 1- ;byte integer into B ;register
32761	0		;1-byte integer
32762	253	ADD IY,BC	;Add without carry ;contents of BC ;register to ;contents of IY ;register
32763	9		;results stored in ;IY register
32764	35	INC HL	;Increment HL ;register by 1
32765	24	JR 32750	;Jump ;unconditionally to ;program counter + ;displacement
32766	239		;Displacement factor

EZRA GROUP II
EZRA GROUP II

LOW LOW LOW LOW LOW LOW LOW LOW LOW
PRICES!
Biorhythms 8K ROM/1K&up.....1.00
Graphics Billboard 8/1up.....1.00
Horse Race 8/1up.....1.00
SPINNER TM(like Rubik's)8/16...2.00
Improved SLOW PAUSE.....1.00
Linear Regression 8/1up.....2.00
CHAR. Generator Demos 8/1up....2.95
SPLIT-A-STACK TM SLOW/1up.....2.00
CHEWTER TM(like PACMN)SLOW/1up.2.95
ISLAND SQUARES TM SLOW/2up.....2.00
Random MUSIC!SLOW/1up.....2.00
BASIC Keyword Demos 8/1....from .50
ORDER,SASE,reSASE gets you GOO-
DIES Catalog and FREE! Program

EZRA GROUP II
EZRA GROUP II

POB 5222 San Diego,California 92105

Software city

THE PROGRAM DISCOUNT CENTERS

We know software for personal computers because software is our ONLY business.

Thousands of programs.
In stock. At discounted prices.

ARMONK, NY, 146 Bedford Rd.(914)273-3677
PINE BROOK, NJ, 101 Route 46 East(201)575-4574
MONTVALE, NJ, 147 Kinderkamack Rd.(201)391-0931
TEANECK, NJ, 161 Cedar La.(201)692-8298
SUMMIT, NJ, 5 Beechwood Rd.(201)273-7904
FAIRVIEW, NJ, 251 Broad Ave.(201)943-9444

Stores Opening Soon:
Green Brook, NJ Morristown, NJ Hartsdale, NY

FRANCHISES AVAILABLE For retail stores. $25-35,000 estimated total investment. Box 313, Closter, NJ 07624

All Stores Open Tues. thru Sat. and Thurs. Evening
Franchises Available/Box 313, Closter, NJ 07624
©1982 America's Number One Software Dealer Inc.

HEAR THE POWER OF YOUR SINCLAIR®

It's been said that the future belongs to those who understand computers. If that's true, what better way to do this than by having them talk to you.

Now you can. Quickly. Simply. Economically.

Let the power of your Sinclair ZX80/81™ or Timex Sinclair 1000™ speak for you now. Through R.I.S.T. Computer Components, Inc. CHATTERBOX™ — your ZX80/81 or TS1000 can speak...clearly, concisely and without limit.

R.I.S.T. has developed a state of the art voice synthesis module *you plug in.* CHATTERBOX utilizes allophone concatenation to synthesize speech, with limitless vocabulary. Sentences, and any English language word can be created.

Best of all, CHATTERBOX is affordable. For $69.95, your ZX80/81 or TS1000 can give you the voice of the future...*today*!

R.I.S.T. Computer Components, Inc.
P.O. BOX 499 , Ft. Hamilton Station , Brooklyn, NY 112090499

My computer is ☐ ZX80* ☐ ZX81 ☐ TS 1000

Enclosed is $ _____ for _____ module(s) at $69.95 ea., plus $3.00 Shpg ea. (includes complete documentation).

☐ *9V Power Supply (optional) $9.95 ea. (Required for ZX80 or expanded system operation).

For orders outside continental U.S., please send bank draft.

NAME _____

ADDRESS _____ APT. _____

CITY _____ STATE _____ ZIP _____

by William Wentz, Rio Rancho, NM

MC Programs — Where and How To Load Them

Machine code (MC) programming offers advantages over programming in BASIC. Quite apart from the issue of learning to program, machine code programmers should consider the various ways of loading machine code (MC) into their computers.

First, *where* to put the MC in memory? The location must meet two requirements: it must be safe from being overwritten by the BASIC system and its location must remain fairly fixed. The ZX81 manual suggests three possible areas of memory. Each has its advantages and disadvantages.

Second, *how* to load the MC into memory? Fundamentally, the process of loading a machine language program consists of POKEing each MC decimal value into a memory address. The entire program occupies consecutive memory locations. This process is usually accomplished by a BASIC program called a *loader*. BASIC loader programs can range from the very simple to rather complex. The most advanced way of loading machine language programs is with an *assembler*. An assembler program directly translates and loads the *mnemonics*, or instruction codes, you type in. These instruction codes do not resemble the almost-English BASIC commands; instead they use abbreviations such as LD for load and EX for exchange.

Here are some thoughts on the *where* and *how* of MC loading for ZX/TS computers.

Program Storage Area

POKEing MC directly into the program storage area will not work. You have to store the MC within a line of a BASIC program, preferably the first line to easily locate the USR address. The BASIC REM statement is probably the best choice because ZX/TS BASIC ignores the REMark statement other than reproducing it in a program listing (i.e., a REM statement is not executed).

If you have thoroughly debugged your machine code and integrated it into a BASIC program, then storing the MC within a REM statement is your best bet. This method uses a minimum of memory space and gives you a constant USR address. Your machine code routine SAVEs on tape right along with the BASIC program.

Once written, the combined BASIC and MC program are useable on any size memory equal to or larger than required for the original program. Also the MC portion of the program need not be *relocatable*. Relocatable means the MC program does not contain any JUMPs, CALLs or LDs to addresses within the MC program itself.

Although not insurmountable, problems associated with using a REM statement for MC storage exist. A mistake in the loaded REM statement is not easy to correct and proofreading the REM statement is almost impossible. Another problem is caused by the machine code 118 dec. (the ZX/TS's code for end-of-line). Using this code within an MC program can cause problems when you LIST the program.

Top of Memory Area

To use the top of memory for MC storage you have to POKE the system variable RAMTOP (Top of memory + 1) with a new address. Then enter NEW to reset the computer. This process gives very safe MC storage at the top of memory. Once you store your MC above RAMTOP, you can only destroy it by indiscriminately POKEing the area, resetting the system (RST 00) or turning off the power to the computer.

Either manually POKE the machine code or use a loader to directly load MC into the top of memory. The USR address will be the address into which you POKE the first executable MC byte.

The top of memory or "above RAMTOP" is an excellent location for machine code utility programs such as line renumbering or Syntactic Sum. Once you load the MC program, you can delete your loader program. Now you can load or enter another BASIC or MC program without interfering with the MC at the top of memory. They will both work fine.

The major drawback to using the top of memory for storage is that this area is not SAVEd on tape. Also, if you want to use your program on a memory size other than that it was written for, it must be relocatable.

Variables Storage Area

You cannot conveniently POKE machine code directly into the variables storage area. Though not specifically mentioned in the ZX/TS manual, using the variables storage area is implied by the suggestion of placing the MC within a string. One method of storing MC this way uses the BASIC LET statement, i.e., LET A$ = ".....". For ease in placing the MC into the string and locating the USR address, put the LET statement containing the MC first in the variables storage area.

Although SAVEd on tape, the variables area is probably the least desirable location. It uses more memory than the others and moves about. If your computer has less than 3.75K of memory, the display file changes as it is used, thus changing the location of the variables. If the variables storage area moves during MC program execution, disaster results. Also, if you delete or change your loader program, you must re-establish the USR address to call the MC.

BASIC Loader Programs

Each BASIC loader program differs, depending on the area you choose to store the machine code and your own requirements.

Should you enter the machine code into the loader program in decimal or hexadecimal notation? While ZX/TS BASIC functions USR and PEEK and the statement POKE use decimal notation, you should definitely consider using hex for machine code programming. Many manuals on Z80 machine language programming and a number of texts on the ZX/TS 8K ROM (such as Dr. Logan's *Understanding Your ZX81 ROM*) use hex notation exclusively.

Here are examples of BASIC loader programs for each of the areas discussed:

REM Statement Loader

Frank O'Hara's unique program *Spaces in REM Statements* (SYNTAX newsletter, Aug. 82) greatly simplified the three step process of loading MC into a REM statement. Making a few changes to *Spaces in REM Statements*, the program becomes a REM Hex Loader or a REM Decimal Loader. See Listings 1 and 2.

Top of Memory Loader

Roderick McConnell's program *Syntactic Sum Loader* (SYNTAX newsletter, Aug. 82) is itself a great loader. His line 1 REM statement does away with the often forgotten step of POKEing a new RAMTOP and then executing a NEW. With a few changes to eliminate some of Rod's restrictions, *Syntactic Sum Loader* becomes a Top of Memory Loader. See Listing 3.

Variables Storage Area Loader

David Ornstein's program *XFER* (SYNTAX newsletter, Jun. 82) shows how to use the variables storage area. His program could be considered a Variables Hex Loader.

Listing 1 REM Hex Loader V 1.0

```
  10 REM 12E■RNDLN ■■TAN
  20 LET H$=".....
  40 LET N=LEN H$/3
  50 LET D=PEEK 16396+256*PEEK 1
6397-2
  60 FOR I=1 TO N
  70 POKE 16515,INT ((D+I)/256)
  80 POKE 16514,D+I-256*PEEK 165
15
  90 RAND USR 16516
 100 POKE D+I,16*CODE H$(I*3-2)+
CODE H$(I*3-1)-476
 110 NEXT I
 120 POKE D-1,INT ((N+2)/256)
 130 POKE D-2,N+2-256*PEEK (D-1)
 200 REM
SYNTACTIC SUM:   20307, 8K ROM
```

(1) The keystrokes in line after 10 REM are: number 1 number 2 letter E graphic-shift W token RND token LN inverse period graphic-shift D token TAN. Lines 10, 50, 60, 70, 80, 90, 110, 120 and 130 are identical to Frank O'Hara's program *Spaces in REM Statements* from the SYNTAX newsletter, Aug. 82.

(2) Type the hex machine code you want to load into line 20 as two-digit hex numbers. For example, type 20 LET H$="3E 10 06" and so on. Put a space after each byte of hex code. No extra spaces, no space after the first quotation mark. Don't forget the space between the last byte and the quotation mark at the end.

(3) If you have a long MC program, you can concatenate H$ for ease in editing. For example, you could enter Dr. Miller's program *Machine Code Screen Fill* from the SYNTAX newsletter, Aug. 82 in hex as follows:

```
  20 LET H$="3E 80 06 04 0E B0 D
? "
  21 LET H$=H$+"0D C2 88 40 05 "
  22 LET H$=H$+"C2 86 40 C9 "
```

(4) Line 200 should contain the REM and nothing else before you run the program. Now RUN the program. Look at the listing. The hex code you typed in line 20 is now in line 200.

(5) Delete lines 10-130. Then you can edit the line number of the 200 REM containing the MC program to make it the first line number of your program.

(6) The USR address of the new MC program is 16514.

(7) The Syntactic Sum printed here is for listing 1 exactly as it appears, with 5 periods instead of hex code in line 20.

Listing 2 REM Decimal Loader V.1.2

```
  10 REM 12E■RNDLN ■■TAN
  20 LET D$=".....
  40 LET N=LEN D$/3
  50 LET D=PEEK 16396+256*PEEK 1
6397-2
```

Continued Next Page

```
  60 FOR I=1 TO N
  70 POKE 16515,INT ((D+I)/256)
  80 POKE 16514,D+I-256*PEEK 165
15
  90 RAND USR 16516
 100 POKE D+I,VAL D$(I*4-3 TO I*
4-1)
 110 NEXT I
 120 POKE D-1,INT ((N+2)/256)
 130 POKE D-2,N+2-256*PEEK (D-1)
 200 REM
SYNTACTIC SUM:   19287, 8K ROM
```

(1) See notes 1, 4, 5 and 6 of listing 1.

(2) Type the decimal machine code to be loaded into line 20. For example, 20 LET D$ = "062 128 006" and so on. Type each decimal MC instruction as a three digit number. For example, enter 2 as 002, 0 as 000. Each three digit number has a single space after it. No extra spaces. No spaces after the quotation mark, and don't forget the space after the last three-digit number and before the quotation mark at the end.

(3) If you have a long MC program, you can concatenate D$ for ease in editing. For example, enter Dr. Miller's program *Machine Code Screen Fill* in decimal as follows:

```
  20 LET D$="062 128 006 004 014
 176 "
  21 LET D$=D$+"215 013 194 136
064 "
  22 LET D$=D$+"005 194 134 064
201 "
```

(4) The Syntactic Sum printed here is for listing 2 exactly as listed, with 5 periods in line 20 and no hex code.

Listing 3 Top of Memory Hex Loader V.1.6

To generate line 10 of listing 3, type in this loader program:

```
  10 REM 12345678901234567890123
4567890123456789012345678 90
  20 FOR X=16514 TO 16563
  30 INPUT N
  40 POKE X,N
  50 NEXT X
SYNTACTIC SUM:   5167, 8K ROM
```

Now RUN your loader program and input the following numbers, one at a time, pressing ENTER or NEWLINE after each. Read the table from left to right and top to bottom.

175	205	35	15	42	4	64	237	114	68	77
3	57	237	91	123	64	237	82	235	25	237
184	237	75	123	64	42	4	64	237	66	34
4	64	42	2	64	237	66	34	2	64	237
98	57	237	66	249	201					

Press LIST and ENTER and line 10 now looks like line 10 of listing 3 following. Input lines 20-150.

```
  10 REM ■LN 7?E■RND GOSUB ???■T
 GOSUB ??RND GOSUB ? FOR ■ GOSUB
 ■ GOSUB ??RNDE■RND GOSUB PI6■RN
DE■RND GOSUB PI6■RND GOSUB ?T GO
SUB PI RAND TAN
  20 LET H$="....."
  40 LET L=LEN H$/3
  50 LET M=L
  60 IF L<20 THEN LET M=20
  70 POKE 16507,M
  80 PRINT USR 16514;" BYTES SAV
ED"
  90 LET N=PEEK 16388+256*PEEK 1
6389
 100 PRINT "RECORD USR ADDRESS "
 N
 110 FOR X=3 TO L*3 STEP 3
 120 POKE N,16*CODE H$(X-2)+CODE
 H$(X-1)-476
 130 LET N=N+1
 140 NEXT X
 150 PRINT " MACHINE CODE LOADED
"
SYNTACTIC SUM:   23944, 8K ROM
```

(1) The code 205,35,15 (Call 0F35) has been added to the line 1 REM statement of *Syntactic Sum Loader* to automatically run the program in FAST mode.

(2) Notes 2 and 3 of listing #1 apply. To enter Dr. Miller's Screen Fill program as an example, we must change it slightly to make it relocatable in RAM. Enter line 20:

20 LET H$="3E 80 06 04 0E B0 D
7 0D 20 FC 05 20 F7 C9 "

Note that you need a space after each two-digit hex code, including the last one. You can enter this routine as two or more lines, concatenating as before, to make the lines easier to edit in case you make a mistake.

(3) The length of the hex machine code in line 20 will be used by the loader program to set the new RAMTOP address.

(4) Line numbers 50 and 60 let you load short MC programs (less than 20 bytes) without causing a computer crash.

(5) This loader can be used with any size memory.

(6) Always jot down the USR address the program gives you because it will not be constant.

(7) To delete the loader after the MC has been located in the top of memory, just enter NEW.

(8) If you have already stored an MC program in the top of memory, the loader can then enter a second MC program just below the first.

Try It

Whether or not you can program in machine code, you can use MC programs on your ZX/TS computer. Try these three methods and choose the one that works most conveniently for the MC you want to run. **SQ**

by David R. Rowland, Montclair, NJ

8K ROM/16K RAM

ZX/TS Directory Program

Directory program stores up to 90 names, addresses, and telephone numbers for later recall and use. With this program, you have several ways to list and call up entries. You can list entries by state, by partial or complete zip codes, by partial or complete names or by coded entries. For example, SYNTAX would appear in a list if 0, 01, 014, 0145 or 01451 were entered as a partial zip or if you were listing by partial or complete name (S, SYN, etc.). You can delete whole entries or make small changes in one step. Directory is written for 90 names; about the limit for a C-10 tape. To change the length, alter lines 1, 2, 3, 101 and 102 accordingly, and use a longer cassette.

To set up the program, enter the listing and RUN. After entries have been made, always use GOTO 10.

Adding and Coding Entries

The program is user-friendly when entering names. To enter state and zip code type in the two-letter abbreviations (ENTER) and then the zip code. To enter phone numbers, type in the area code (ENTER), the exchange (ENTER), and the four digit number. If certain data are unknown, simply press ENTER to leave the item blank.

Three examples of codes are included in the program, on lines 882 to 889. You can set up your own code system, using numbers from 0 to 9, or letters A to Z. Up to four codes can be entered for each name. When entering codes, type in each value followed by ENTER, and continue to press ENTER until the program resumes.

Changing Entries

To change data, the entire line has to be reentered; for example, full street address even if only changing the house number. This applies to all items, including codes. To change the phone number, type in the full number, including hyphens, before ENTERing.

To exit routines use STOP. To save changes or additions, a user-friendly save routine is provided in the menu. **SQ**

```
1 DIM N$(1,31)
2 DIM S$(1,31)
3 DIM T$(1,31)
```

```
   5 REM START PROGRAM WITH "GO
TO 10"AFTER INITIAL RUN
   9 LET D=0
  10 PRINT AT 0,12;"HELLO";AT 2,
2;"ENTER CODE FOR WHAT IS WANTED
"
  11 PRINT AT 4,0;"1.  ENTER MOR
E NAMES"
  12 PRINT "2.  CHANGE SELECTED
NAME OR       OTHER DATA"
  13 PRINT "3.  CHECK ALL ENTRIE
S"
  14 PRINT "4.  LIST ALL NAMES B
EGINNING",,"  WITH A LETTER"
  15 PRINT "5.  LIST ALL NAMES I
N STATE"
  16 PRINT "5.  LIST ALL NAMES N
OT IN STATE"
  17 PRINT "7.  LIST ALL NAMES B
Y ZIP CODE"
  18 PRINT "8.  LIST ALL CODED N
AMES"
  19 PRINT "9.  SAVE REVISED DIR
ECTORY"
  20 INPUT J
  25 CLS
  30 IF J<=9 THEN GOSUB J*100
  35 CLS
  40 GOTO 10
  60 PRINT AT 8,0;"CALL UP PROGR
AM WITH ""LIST"",AND DELETE LINE
S 1 TO 5,9,60,61,62,65+101."
  61 PRINT ;"THEN RESUME USING
""GOTO 10""."
  62 PRINT ;,"DO NOT USE ""RUN""
[redacted]"
  65 STOP
 100 FOR L=D+1 TO 90
 101 IF L=1 THEN GOTO 60
 102 IF L=91 THEN GOTO 196
 105 PRINT AT 4,0;"TYPE IN NAME
(LAST FIRST)"
 110 INPUT N$(L)
 112 LET D=D+1
 115 PRINT AT 6,0;"STREET ADDRES
S"
 120 INPUT S$(L, TO 19)
 125 PRINT AT 8,0;"CITY"
 130 INPUT T$(L, TO 19)
 135 PRINT AT 10,0;"STATE + ZIP"
 140 INPUT T$(L,23 TO 25)
 141 INPUT T$(L,27 TO )
 142 CLS
 145 PRINT AT 4,0;"PHONE NO., WI
TH AREA CODE"
 150 INPUT S$(L,20 TO 22)
 151 LET S$(L,23)="-"
 152 INPUT S$(L,24 TO 26)
 153 INPUT S$(L,28 TO )
 154 LET S$(L,27)="-"
```

Continued Next Page

```
 155 PRINT AT 6,0;"ANY SPECIAL C
ODES? Y OR N."
 160 INPUT J$
 165 CLS
 170 IF J$="Y" THEN GOSUB 750
 172 PRINT AT 0,0;L,,N$(L),S$(L)
 T$(L),,,
 175 PRINT "ALL CORRECT?  Y OR N
"
 178 INPUT J$
 180 IF J$<>"N" THEN GOTO 184
 181 LET A=1
 182 GOSUB 330
 183 LET A=0
 184 CLS
 185 PRINT AT 6,0;"ANY MORE NAME
S?  Y OR N."
 187 INPUT W$
 190 CLS
 192 IF W$<>"Y" THEN RETURN
 195 NEXT L
 196 PRINT "[illegible]"
 197 PRINT AT 20,0;"PRESS ""[illegible]
"" TO REGAIN MENU"
 198 STOP
 199 RETURN
 200 PRINT AT 3,0;"ENTER REGISTE
R NO.  OF NAME TO BE CHANGED."
 205 INPUT L
 210 PRINT L,,N$(L),S$(L),T$(L),
,,
 215 LET A=1
 220 GOSUB 330
 230 RETURN
 300 FOR L=1 TO D
 301 LET A=0
 305 PRINT AT 0,0;L,,N$(L),S$(L)
,T$(L),,,
 310 PRINT "ARE NAME+ADDRESS COR
RECT? Y OR N"
 315 INPUT J$
 320 IF J$="N" THEN GOTO 330
 322 CLS
 324 IF A=1 THEN RETURN
 326 NEXT L
 328 RETURN
 330 PRINT ,,"WHAT LINE IS WRONG
?",,,"1.  NAME",,"2.  STREET ADD
RESS",,"3.  CITY",,"4.  STATE",,"
5.  ZIP",,"6.  PHONE NO.",,"7.
CODE"
 331 PRINT "8.  NOTHING",,"9.  E
VERYTHING - DELETE ENTRY"
 340 INPUT Q
 350 IF Q=1 THEN INPUT N$(L)
 355 IF Q=2 THEN INPUT S$(L, TO
19)
 360 IF Q=3 THEN INPUT T$(L, TO
19)
 365 IF Q=4 THEN INPUT T$(L,23 T
O 25)
 370 IF Q=5 THEN INPUT T$(L,27 T
O 31)
 375 IF Q=6 THEN INPUT S$(L,20 T
O )
 380 IF Q=7 THEN GOSUB 750
 385 IF Q=8 THEN GOTO 35
 386 IF Q=9 THEN GOSUB 660
 389 CLS
 395 GOTO 305
 400 PRINT AT 4,0;"ENTER LETTER(
S) OR NAME TO BE","LOCATED"
 405 LET Q=0
 410 INPUT W$
 420 CLS
 430 FOR L=1 TO D
 440 IF W$=N$(L, TO LEN W$) THEN
 GOSUB 465
 450 NEXT L
 455 IF Q=0 THEN PRINT AT 7,10;
[illegible]"
 457 PRINT AT 20,0;"PRESS ""[illegible]
"" TO REGAIN MENU"
 460 STOP
 462 RETURN
 465 PRINT L,,N$(L),S$(L),T$(L)
,,
```

```
 470 LET Q=Q+1
 480 IF INT (Q/4)=Q/4 THEN GOTO
490
 485 RETURN
 490 PRINT AT 20,0;"PRESS ""[illegible]
"" TO RESUME LISTING"
 492 STOP
 495 RETURN
 497 GOTO 450
 500 PRINT AT 5,0;"ENTER TWO LET
TER STATE CODE"
 505 LET Q=0
 510 INPUT W$
 515 CLS
 520 FOR L=1 TO D
 525 IF W$=T$(L,23 TO 24) THEN G
OSUB 455
 530 NEXT L
 540 GOSUB 455
 550 RETURN
 600 PRINT AT 5,0;"ENTER TWO LET
TER STATE CODE"
 605 LET Q=0
 610 INPUT W$
 615 CLS
 620 FOR L=1 TO D
 625 IF W$<>T$(L,23 TO 24) THEN
GOSUB 465
 630 NEXT L
 640 GOSUB 455
 650 RETURN
 660 LET N$(L)=N$(D)
 665 LET S$(L)=S$(D)
 670 LET T$(L)=T$(D)
 675 LET N$(D)=""
 680 LET S$(D)=""
 685 LET T$(D)=""
 690 LET D=D-1
 695 RETURN
 700 PRINT AT 5,0;"ENTER 5 NUMBE
R ZIP"
 702 LET Q=0
 704 INPUT W$
 706 CLS
 708 FOR L=1 TO D
 710 IF W$=T$(L,27 TO (26+LEN W$
)) THEN GOSUB 465
 712 NEXT L
 714 GOSUB 455
 716 RETURN
 750 CLS
 751 GOSUB 680
 765 FOR P=18 TO 21
 770 INPUT T$(L,P)
 775 NEXT P
 785 CLS
 790 RETURN
 800 LET Q=0
 801 CLS
 805 GOSUB 881
 810 INPUT A$
 812 CLS
 815 FOR L=1 TO D
 820 FOR P=18 TO 21
 825 IF T$(L,P)=A$ THEN GOSUB 46
5
 830 NEXT P
 833 NEXT L
 845 GOSUB 455
 850 RETURN
 880 PRINT L,,N$(L),S$(L),T$(L),
,,
 881 PRINT AT 4,10;"CODE CHOICE"
,,,
 882 PRINT "1.  FAMILY MEMBERS"
 883 PRINT "2.  BUSINESS NUMBERS
"
 889 PRINT "9.  CHRISTMAS CARDS"
 890 RETURN
 900 PRINT ,,"IF""SAVE""IS WANTE
D, ENTER ""1""."
 910 INPUT E
 915 IF E=1 THEN SAVE "DIRECTOR[illegible]
"
 917 CLS
 920 GOTO 10
SYNTACTIC SUM:  34371, 8K ROM
```

by David M. Straub, 500 Eastwood Drive, Petaluma, CA 94952

Keyboard Conversions For ZX/TS

Your ZX/TS computer's membrane keyboard offers many advantages. But many people prefer the convenience of a bigger keyboard. We'll look at what's available in add-on keyboards and some of the costs involved to help you shop around.

When you add a bigger keyboard to your ZX/TS computer, you have three basic choices:

- Pre-assembled—the keyboard comes completely wired; you need only plug it in and start typing. Some require soldering and permanent removal of the membrane keyboard connectors.
- Semi-assembled—the keyboard is wired, but you must also wire an interface circuit between the keyboard and your computer.
- Non-assembled—you must wire both the keyboard (hooking two wires to each key) and the interface circuit.

In choosing a big keyboard to add to your ZX/TS computer, first decide what kind, if any, of options or extras you want. These extras may include a numeric keypad (an extra set of number keys laid out like a calculator), reverse video, automatic SHIFTing, or a longer ribbon cable between the keyboard and the computer. These options usually mean your keyboard will cost you more in assembly time (for kits) or money (for pre-assembled units).

Next, consider these questions:

1. How much time do you have to put the keyboard together?
2. How much money do you want to spend?
3. How much skill (or patience) do you have in soldering or wire-wrapping?

How long will it take you?

Two factors affect the amount of time you'll need to attach the keyboard to your computer: which keyboard you buy and your soldering or wire-wrapping skills. Many pre-assembled models hook up in 5 minutes or less. Semi-assembled models can take anywhere from 1 to 3 hours. The time varies more with non-assembled keyboards. They can take as little as 4 hours and as long as 8. Plan to spend 1/4 to 1/3 of your total time debugging, or finding the errors in your work. One way to minimize debugging time is to work slowly and carefully, reducing the chance of errors that can be very hard to find later.

What will you spend?

Obviously, the more extras you get and the less work you have to do, the more the unit will cost. Pre-assembled keyboards go for $70-100. Semi-assembled units run between $40 and $60, and I've seen non-assembled keyboards for as low as $15. Surplus keyboards can cost as little as $5.

What expertise do you need?

If you don't know how to solder or don't feel comfortable doing it, look for an unassembled kit you can wire-wrap. Wire-wrapping is easier and not as messy as soldering. If you don't know how to wire-wrap or you don't have the time for either soldering or wire-wrapping, spend the extra money to invest in a pre-assembled unit. To hook up a pre-assembled keyboard you usually need only be able to connect two plugs.

Other considerations

When shopping around for a keyboard, make sure you can rely on the company for technical help by mail or phone should you need it. Does the company guarantee their work? If so, what does the guarantee cover and for how long?

What exactly comes with your kit? Check this out before buying so you know what parts you must buy extra (and pay for extra). Does the ribbon cable have a detachable plug so you can remove the keyboard without desoldering?

Where do you start looking? First try ads in computer magazines. Several companies (see accompanying list) offer assembled and kit keyboards specifically for ZX/TS computers. If you prefer a do-it-yourself approach, the

least expensive keyboards are usually found at electronics surplus houses, but of course, they require the most work.

After finding a company that appears to have what you want, feel free to write and ask any questions. If the company is legitimate, they will go out of their way to help you. Buying things through the mail is sometimes a risky thing. When you are dealing with a relatively large sum of money (up to the cost of the computer itself!), you should take every precaution to ensure that you get your money's worth.

Interfacing a Non-Assembled Keyboard to Your ZX/TS

So you decided to assemble your own keyboard. We'll go through each step to hook up your new keyboard. To start, look at Table 1 to get an idea of the tools and components you'll need. If you bought a kit, the manufacturer will have provided some of these.

Make sure you **read all the directions before you even pick up a soldering iron.** You must be thoroughly familiar with the information before you start. Not all keyboards will require all steps described here; if you got a keyboard kit specifically for ZX/TS computers, you can skip some of the steps following.

Table 1

Item	Amount	Function
14-pin single ended DIP jumper	1	Hook keyboard to computer
14-pin DIP socket	1	Goes between the jumper and the keyboard
ZX81 keypads	1 set	Customize the keyboard with ZX81 keywords
Case	1	House the keyboard
Mounting brackets	2	Support the keyboard
Wire-wrap wire	5 feet	Wiring the keyboard
Sealant	1 can	Protecting keytops
Solder	1 roll	Wiring the keyboard
Spray paint	1 can	Painting keys

Tools

Wire-wrap tool
Fine-tip soldering iron
Wire cutters/strippers
Sander and sandpaper
Screwdrivers (Phillips and blade)
Ohmmeter
Needle nose pliers
Regular pliers
Desoldering tool (bulb/wick)

1. The keys

If you buy a totally unwired keyboard, choose one with molded keys instead of individual keyswitches. It will simplify your wiring later.

If your keyboard comes without ZX/TS labels on the keytops, you must first relabel them. (Ed. note: Some keyboards, such as late model TRS-80 keyboards, have keys whose tops cannot be removed without damage. Avoid buying this kind.) Carefully remove each keytop from the keyboard. Start on an edge and work in rows. Be careful that none of the inner workings of the key mechanism fly apart and are lost. This is best done by applying steady, constant pressure rather than a sudden jerk or tug.

Take the keys to an area appropriate for spray painting. I suggest using either glossy gray or black spray paint. Give each key a thorough coat, let dry, and do it again.

An easy way to label the keys is to cut out the key labels from a color ZX81 or TS1000 ad in a magazine. Because most large keyboard keys are dish-shaped, choose key labels smaller than the actual keytop or you will not be able to read the legend easily. Give the paper key labels some kind of protection—white glue works well. Let the paper labels dry at least a couple of hours before you touch them.

When the paper labels are dry, attach them to the keytops. DO NOT USE RUBBER CEMENT. Rubber cement will dissolve when you apply your final sealant. Avoid petroleum- or rubber-related adhesives. Use a water-soluble glue like Elmer's. Unless you have a space bar (as opposed to key), larger shift keys or a larger return or enter key, it doesn't matter which key cutout you put on which keytop right now. (Ed. note: Some keyboards are contoured, meaning that the keys have different slopes in different rows. If your keyboard is one of these, arrange them in their correct layout, then match ZX/TS key labels to them.)

After affixing all the cutouts to the keys, put the keys on some newspaper and give them a good coat of spray lacquer. Wait until the keys are dry to the touch (about 15 minutes), then give them another good coat. When thoroughly dry (preferably overnight), the keytops will have an attractive, durable finish that will withstand many keystrokes without smearing or fading.

Reinsert the keytops in their proper sockets. Keep each key in the same position relative to those around it as on the computer's original keyboard. If you have extra keys or want the keys in a non-standard pattern, you will have to make the proper adjustments and keep track of the changes yourself.

2. Wiring the matrix

Turn your keyboard over, key side down. Many keyboards have a printed circuit (PC) board covering wires known as key pins or posts. Remove the PC board, being extremely careful not to damage or bend any pins. If the board is soldered to the posts, use solderwick to draw off all solder from each connection. The PC board may have runs or landings, most commonly called traces (metal lines that connect the pins). If the board was designed for the ZX/TS computer, do not cut the runs, otherwise, you must remove them before doing any soldering.

With an orbital or belt sander (medium grit sand paper), completely sand the PC board surface until no

metal is present on either side of the board. This removes the possibility that the PC board will cause any short-circuit problems. (If you prefer not to remove the epoxy sealant from your PC board, cut the traces to break the circuits. Use a knife or a hobby grinder. At SYNTAX we cut traces and pull them up with a razor blade and long-nose pliers.)

Now study Figure 1. This is the keyboard wiring pattern. You will reconstruct this pattern on your new keyboard. The dots represent pins for each key. Note that you wire columns, then rows of keys, wiring only one post of the pair at a time. Wire heavy lines first. For your convenience, the pattern in Figure 1 is shown in reverse. It appears as the keyboard does when it is key-side down. Study the figure and make sure it makes sense to you before you proceed.

If you have a keyboard with individual keyswitches, wire them together to keep the same spacial relationship between keys. When all the keys are wired together, they must fit back into the holes they came from on the PC board. If the pins are long enough, you may wish to replace the keys on the board and solder onto the part of the pin that strokes through the board.

Warm up your soldering iron. Prepare small leads or jumper wires to go from key to key (wire-wrap is good for this). When you are ready to start (remember, heavy lines first), orient yourself on the keyboard. Find the key you want to start with on the top (key) side, then find the corresponding pins on the reverse. Make one or two tight twists of wire around each post.

Secure the wire to the post with a small dab of solder (a big glob will cause problems later). Do not solder on an upside-down switch. Hot solder is like any other liquid—it will flow down, right into your key. Hold the board with the keys at an angle, pins facing downward. A vise really helps. Now if you have too much solder, it will flow to the end of the pins, not into the keyswitch. Be careful not to use too much wire—you'll have a lot of wires when you're done. Keep things neat so you can easily check your connections if something goes wrong. If you follow these precautions, you should have a good mechanical contact between the posts and the jumper wires.

Run the wires outside the terminals of a single key, not between them. This gives you more room between the pairs of posts. Push the wires as close to the post base as possible because you'll put more than one wire on some posts. Pushing the wires to the base leaves room to replace the PC board if you haven't done so already.

To keep the wire from popping up as you solder, hold it down with a small tool (such as pliers) as it cools. Do not touch the connection, however. Take your time and be patient. If you rush, you may get sloppy.

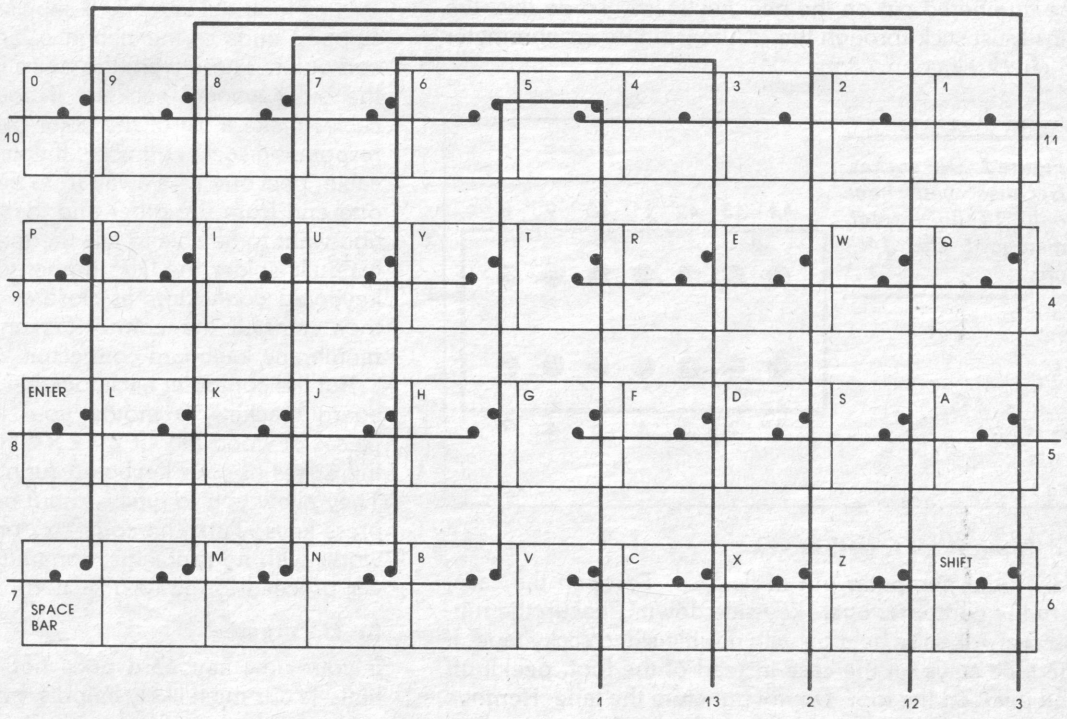


Figure 1. Wire the heavy lines first. Then check all connections. Use black electrical tape to cover places where bare wire is exposed and may be touched by other bare wire. Now solder the light lines (lines 7, 9, 8, 10, 6, 5, 4, 11).

Make sure your solder joints are smooth and flowing. If one doesn't look good, reheat it and try again. A "cold joint" (one without good electrical contact) can give you all kinds of problems and is difficult to track down. You can recognize cold joints by their appearance—they look dull and crystalline. A good "hot" joint is shiny silver and looks flowing. As you solder, clean between each of your connections with a small non-wire brush.

After wiring each column and each half-row, check your wiring with an ohmmeter. When you press a key, you should read little or no resistance on your ohmmeter.

If you have any extra keys, now is the time to wire them. Run jumpers from where the key would normally be to where you want it to be. Use your extra keys for single functions. An extra key is one that is not part of the existing pattern on the ZX/TS keyboard.

If you didn't wire with the board on, replace the PC board over the wired keys. Be very patient in doing this—don't force it or put any extreme pressure on the board. If it won't go on, look for the guilty pins. Remove the extra solder, straighten the pin, or push the wires closer to the base of the pin to clear the way for the PC board. To do this, reheat the solder with your soldering iron. Hold the wire down with a small tool, such as pliers, until the solder cools and hardens. If you just wire-wrapped without soldering, you must rewrap the wire closer to the pin's base. Do not just push it down.

3. Interfacing

Look at Figure 2. Your 14-pin DIP socket (pin side up) has numbers corresponding to Figure 2. Notice the numbers 1-13 point to various keyboard pins. Run jumper wires between each corresponding pin on the socket and the numbered pin on the new keyboard. To do this, the pins must stick through the PC board. Use an ohmmeter to check all connections.

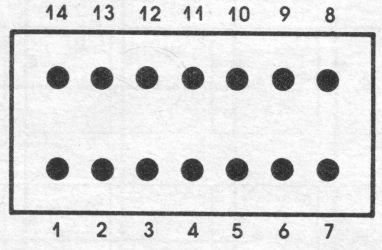


Figure 2. DIP socket. Because you have only 13 lines total, disregard the 14th pin.

4. Tying it all together

Take your computer out of its case. To open the case, turn the computer over, key-side down. Peel up the rubber feet—they're held on with double-sided sticky tape. If the tape stays on the case instead of the foot, peel it off and put it on the foot. Do not puncture the tape. Remove the screws underneath the feet and any others on the bottom of the case. The two halves will come apart.

Now remove the screws holding the board in and disconnect the membrane keyboard, desolder the sockets from the PC board the same way you desoldered the keyboard pins. Strip and tin the end wires of the male jumper cable. Then plug the jumper cable socketed end into the DIP socket wired on the keyboard. Use your ohmmeter to test the socket against the stripped ends of the jumper. This lets you number the stripped ends of the jumper cable exactly like the socket (see Figure 2B). Now stick the stripped end of the jumper cable through the slit in the top of the case where the keyboard ribbon cable from the membrane keyboard used to go.

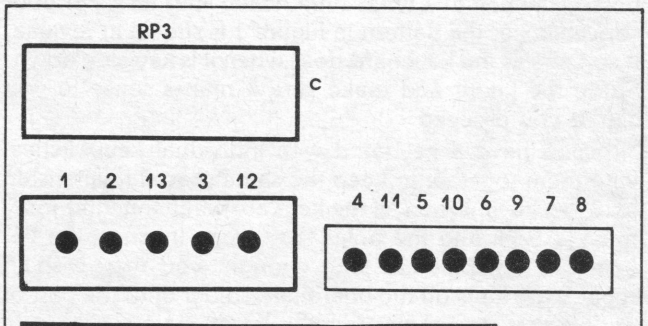


Figure 3. Numbers corresponding to the cable should have the following order 1, 14, 2, 13, 3, 12, 4, 11, 5, 10, 6, 9, 7, 8. The order may be reversed. In that case, merely give the ribbon a half twist. When you are ready to insert the stripped ends of the ribbon cable into the computer, use the order shown above. Remember to skip pin 14.

Now look at Figure 3. It shows the order to solder the stripped ends of the ribbon cable into the holes in the computer. These were the holes left when you removed the old keyboard sockets. If you have colored ribbon cable, make a list of the color of each wire and its corresponding socket number. If you have solid color ribbon cable, peel one wire away from the rest. This lets you tell one end from the other and identify individual wires. If you want to be able to use the membrane keyboard too, carefully solder the DIP jumper wires to the pins on the keyboard connectors as close as possible to the plastic, then reinstall the connectors and replace the flexible membrane keyboard connector.

Put the computer back together. If you don't have keyboard brackets to mount your keyboard on, get two pieces of wood (about 2 × 2 × 4 inches). Put them under the edges of your keyboard running from front to back. They allow you to type without bending the pins as you press keys. Turn the computer on and start typing. If it works with no problems, congratulations. The rest of us will proceed to the next section.

5. Debugging

If your new keyboard does not work, here are a few hints. Your most likely culprits are short circuits and bad solder joints.

- When checking for a short circuit, remember to unplug the keyboard and computer, or you will get erroneous readings.

• If a key or set of keys doesn't work, check the path of wires all the way to the connector inside your ZX/TS computer for bad joints.

Be careful that when you fix one mistake you don't create others. Again, take your time and do things right—it will save you lots of time and headaches. Then if your keyboard doesn't work, you can blame it on a faulty key.

Sources of Pre-Assembled Keyboards

These vendors sell pre-assembled keyboards for ZX/TS computers. Keyboards vary in their features and prices. Some are full typewriter size; others are closer to the original membrane keyboard in size. Some offer full-travel keys, space bars, automatic repeat keys, and separate keys for shifted functions. Most companies also offer keyboard enclosures to hold the keyboard and computer neatly.

Prices and vendors change in this field; check with companies listed below for details before ordering. New vendors are introducing new products all the time. This list is not meant to be exhaustive, and includes only North American sources. To keep up-to-date on new products, check with SYNTAX newsletter. **SQ**

Vendor	Address	Features
E-Z Key	Suite 75 711 Southern Artery Quincy, MA 02169	SB 2S AS KP PL C/NC LD Large Calculator Style
Gladstone Electronics	1736 Avenue Rd. Toronto, Ontario Canada M5M 3Y7 or 1585 Kenmore Ave. Buffalo, NY 14217	SB 2S PL C/NC Small Typewriter Style
Kopak Creations	448 West 55th St. New York, NY 10019	SB 2S PL C/NC LD Medium Typewriter Style
LJH Enterprises	P.O. Box 6305 Orange, CA 92667	SB 2S NPL NLD Small Typewriter Style
Mindware Inc.	15 Tech Circle Natick, MA 01760	SB 2S PL NLD Medium Typewriter Style

Key:
- SB Space Bar
- 2S Two Shift Keys
- AS Automatic Shift on Some Keys
- PL Plug-In
- NPL Not Plug-In: Solder-In
- LD ZX81 Legends on Keys
- NLD Not ZX81 Legends
- C Case
- NC No Case
- KP Numeric Keypad

Using Extra Keys on Big Keyboard—A Simpler Approach

David Ornstein, Newton, MA

Many ZX81 users, dissatisfied with Sinclair's flat membrane-based keyboard, are substituting larger, ASCII-type keyboards. This gives two major advantages. First, you get much more tactile feedback with the more conventional keyboard. Second, you can create special-function keys with the unused keys on the larger keyboard.

Included in these special-function keys are automatically shifted keys. For example, you could have a separate DELETE key instead of using SHIFT 0.

In the Jun. 81 SYNTAX, an article detailed how to set this up. However, that plan had one drawback: it required ICs, power, and a moderate amount of wiring.

This alternate method, though slightly limited, will probably cover most of your SHIFTed desires. With two diodes per key, you can emulate the SHIFTed functions for the following keys: A, Q, 1, 0, P, NEWLINE and SPACE. On an 8K ROM, these SHIFTed keys yield STOP, " ", EDIT, RUBOUT, ", FUNCTION and BREAK.

The circuit for a single key is shown in Figure 1.

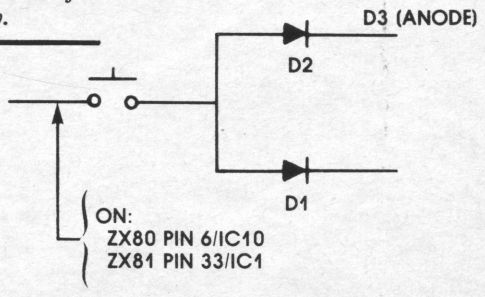
Here is a list of connections for the A side of D1:

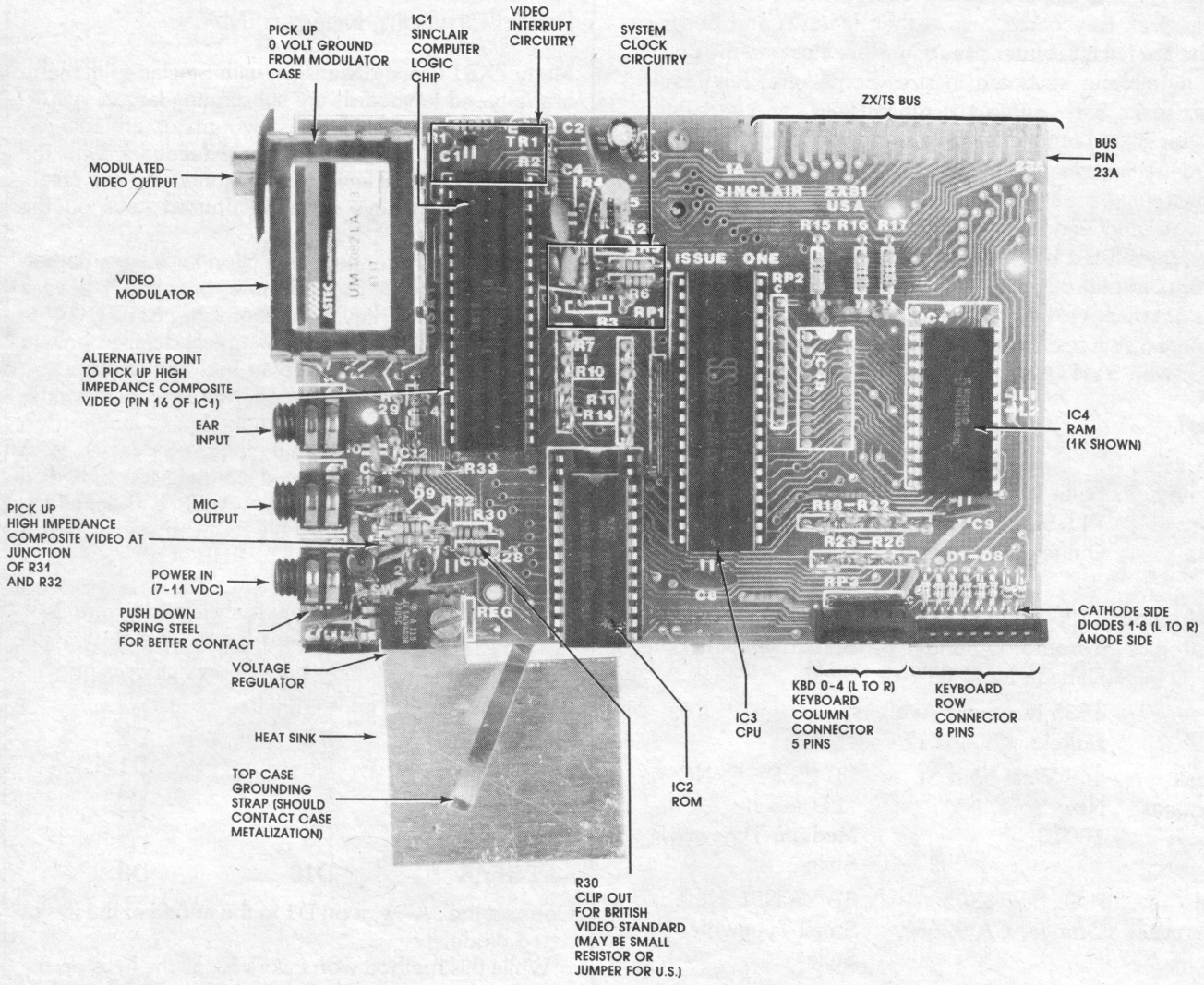
	ZX80	ZX81/TS1000
STOP	D4	D4
" "	D5	D2
EDIT	D6	D1
RUBOUT	D7	D3
"	D8	D5
FUNCTION	D9	D7
BREAK	D10	D8

Connect the "A" wire on D1 to the anode of the designated diode.

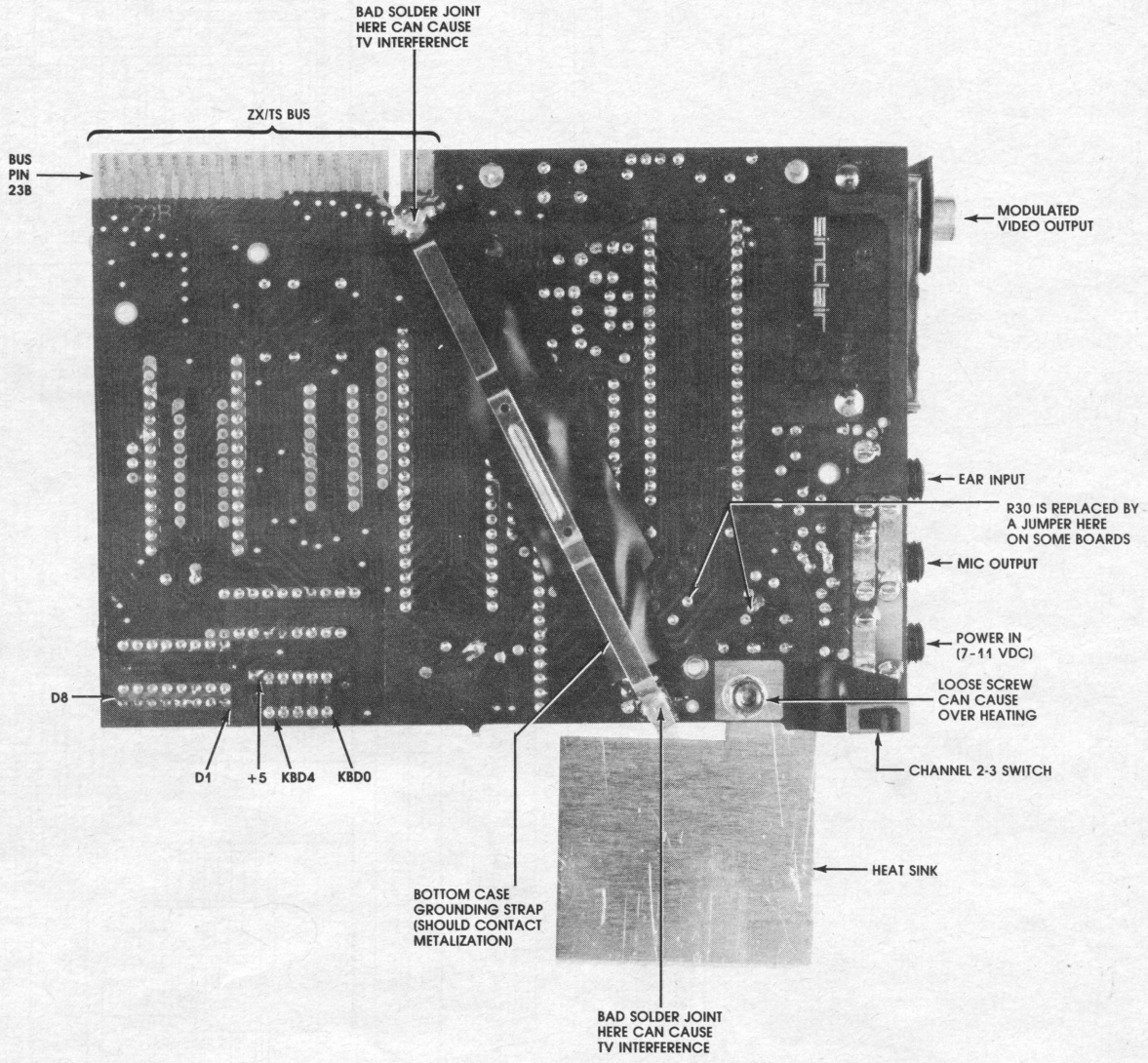
While this method won't work for all the keys on the system, it's certainly easier to implement than a full, IC-based circuit. **SQ**

Figure 1. Circuit for a single key.

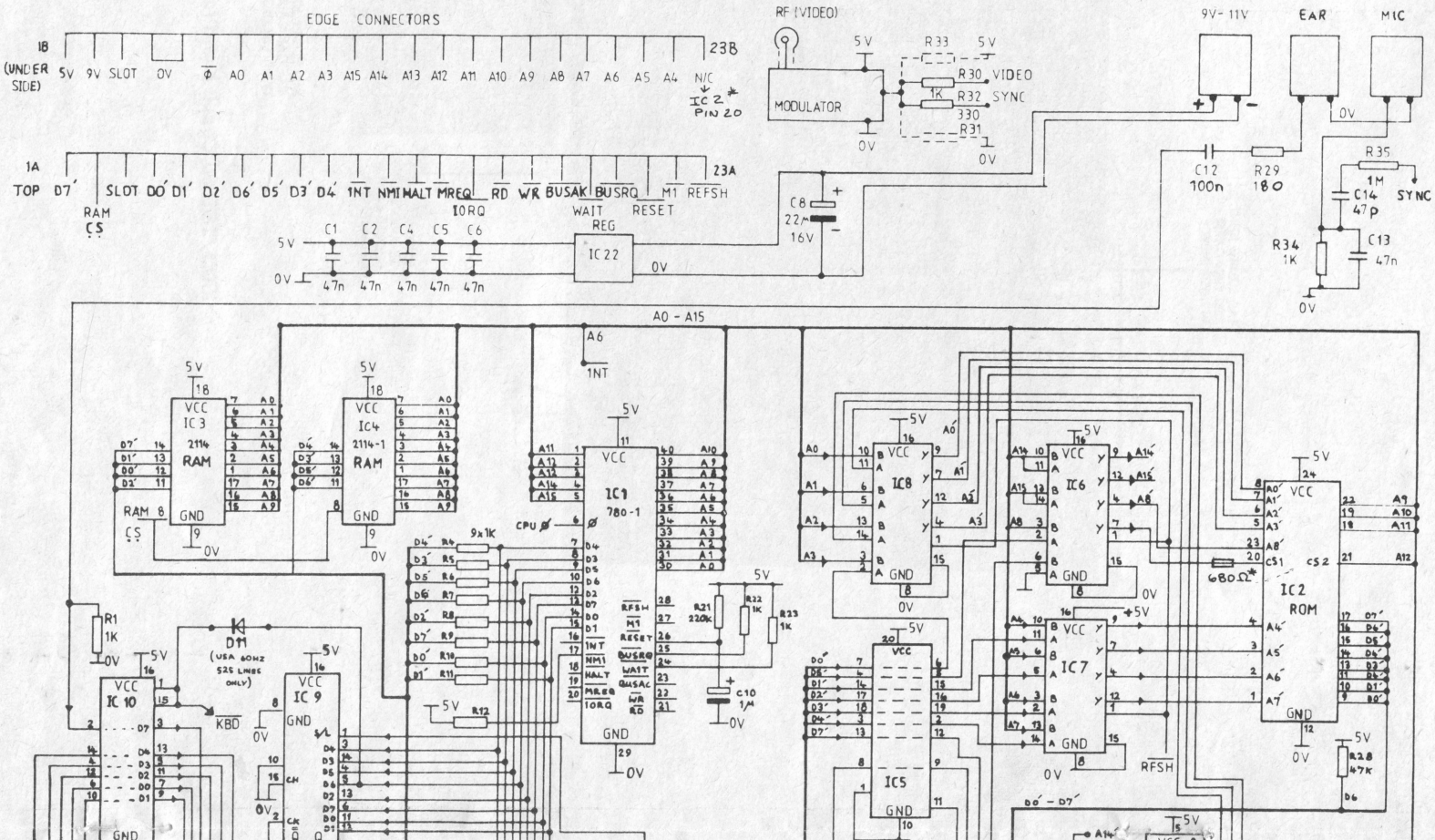


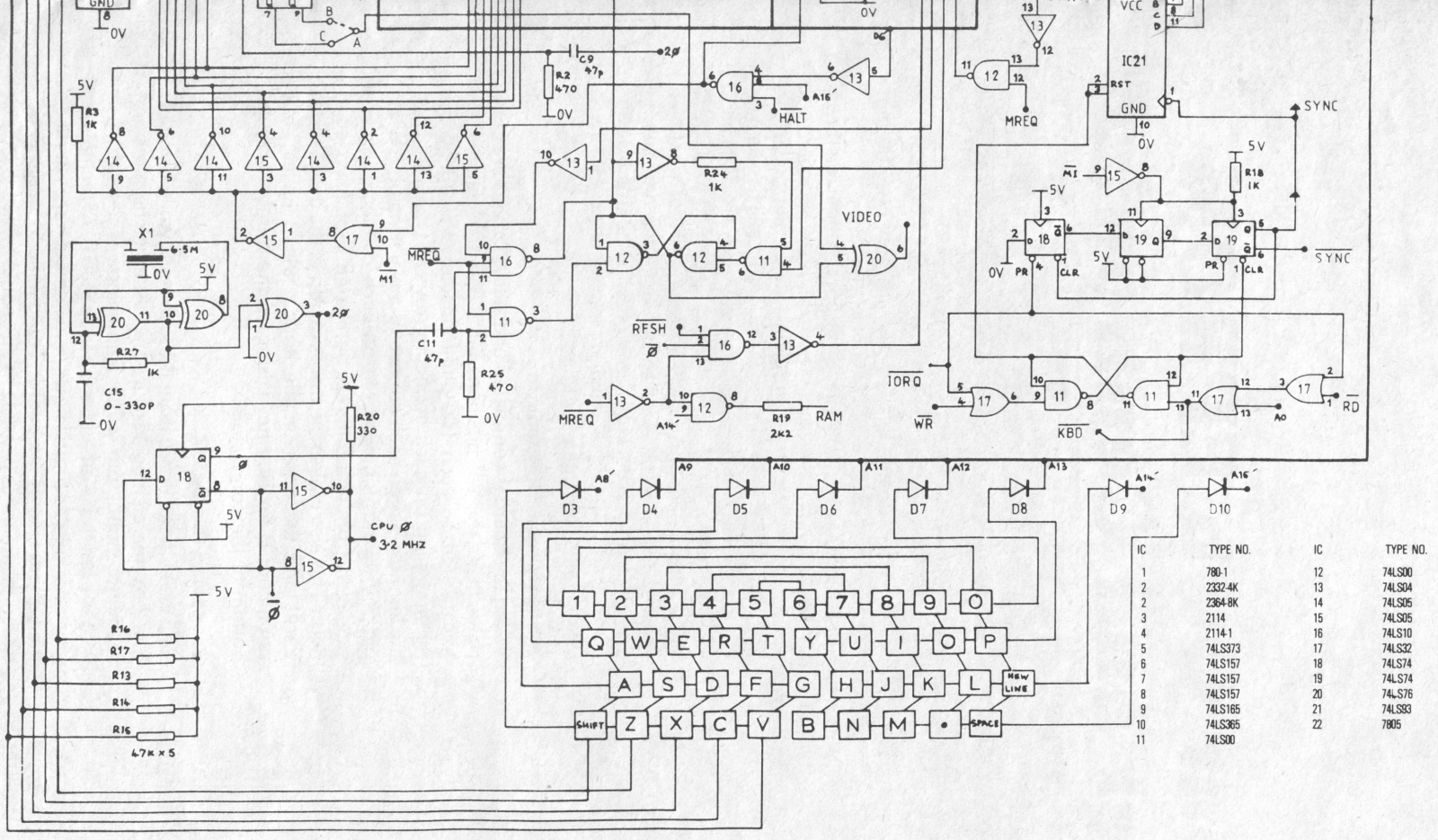


ZX81/TS1000 Circuit board (component side). Use this photograph to locate key items inside your computer. The board shown is from a ZX81.



ZX81/TS1000 Circuit board (underside)


EDGE CONNECTORS
18 (UNDER SIDE) 5V 9V SLOT 0V Φ A0 A1 A2 A3 A15 A14 A13 A12 A11 A10 A9 A8 A7 A6 A5 A4 N/C 23B
IC 2 * PIN 20
1A TOP D7' SLOT D0' D1' D2' D6' D5' D3' D4' INT NMI HALT MREQ IORQ RD WR BUSAK BUSRQ M1 RESET REFSH 23A
RAM CS
RF (VIDEO)
MODULATOR
R33 5V
R30 VIDEO
1K
R32 SYNC
330
R31
9V- 11V
EAR
MIC
C12 100n
R29 180
R35 1M
C14 47p
SYNC
R34 1K
C13 47n
C8 22M 16V
C1 C2 C4 C5 C6 47n
IC 22 REG
WAIT
RESET
A0 - A15
A6
INT
IC3 2114 RAM
IC4 2114-1 RAM
RAM 8
IC1 780-1
CPU
9x1K
R1 1K
D1 (USA 60HZ 525 LINES ONLY)
KBD
IC 10
IC 9
IC8
IC6
IC7
IC5
IC2 ROM
680Ω*
CS1
CS2
RFSH
R21 220K
R22 1K
R23 1K
C10 1M
R12
R28 47K
D0 - D7



IC	TYPE NO.	IC	TYPE NO.
1	780-1	12	74LS00
2	2332-4K	13	74LS04
2	2364-8K	14	74LS05
3	2114	15	74LS05
4	2114-1	16	74LS10
5	74LS373	17	74LS32
6	74LS157	18	74LS74
7	74LS157	19	74LS74
8	74LS157	20	74LS76
9	74LS165	21	74LS93
10	74LS365	22	7805
11	74LS00		

*Mods for Memopak 64K, I/F, CAI/O and ZX-99

by John Oliger, Indianapolis, IN

Build Your Own EPROM Programmer and Centronics Printer Interface—Part I

In these projects, we'll connect and drive a standard ASCII printer with a Sinclair ZX81. Although we will refer primarily to ZX81s, the projects apply equally to Timex-Sinclair 1000s. I have also connected a printer to my ZX80.

These projects are not for beginners. If you have sufficient experience with electronics to take them on, be sure to read the directions completely before starting.

Along the way, we will build:

1. An expansion board.
2. A Centronics-compatible parallel printer port.
3. A 2716 EPROM programmer, programmable from BASIC.
4. Two EPROM read boards, one mapped 8-16K and one mapped 0-6K.

You need 16K or more RAM for these projects.

You can build all of these projects, or you can follow along and only do the ones you prefer. If you do all as shown, your ASCII printer will accept all of the Sinclair printer commands LLIST, LPRINT and COPY. You will still be able to use the ZX printer if you choose.

If you would like to build the projects but not the boards, I can supply bare boards to you:

Expansion board	$18
Parallel printer port board	$10
EPROM programmer board	$10
EPROM read board	$10

Order from John Oliger, 10115 Nassau Lane, Indianapolis, IN 46229.

The Expansion Board

The expansion board is simply a big piece of double-sided, 1/16 inch thick printed circuit board. We'll etch lines into it to put six 100-pin edge connectors on it. Each edge connector holds two boards to give you 12 slots.

You can get the edge connectors from: Poly Paks Inc., P.O. Box 942, South Lynnfield, MA 01940. Catalog number 7119. Price $1.49 each or 3 for $3.00. Write and request a catalog. In catalog number 248-A, the connectors are on page 12. The ad says they are 200-pin, but they are actually 100-pin connectors.

These 100-pin double readout edge connectors have pin spacing of 0.1 inch on center. This is the same dimension used on Sinclair's edge plug. The center pins (pins 2-24 and 27-49) on the expansion board will have a pin-out identical to Sinclair's. Pins 1A and 26A are dedicated to −5V, and pins 1B and 26B are dedicated to +12V. You can use pins 25A, 25B, 50A and 50B for whatever you like. None of these projects require the new supply voltages. They are used with 16K RAM boards I developed using 4116 16K × 1 dynamic RAM chips. But that's another project.

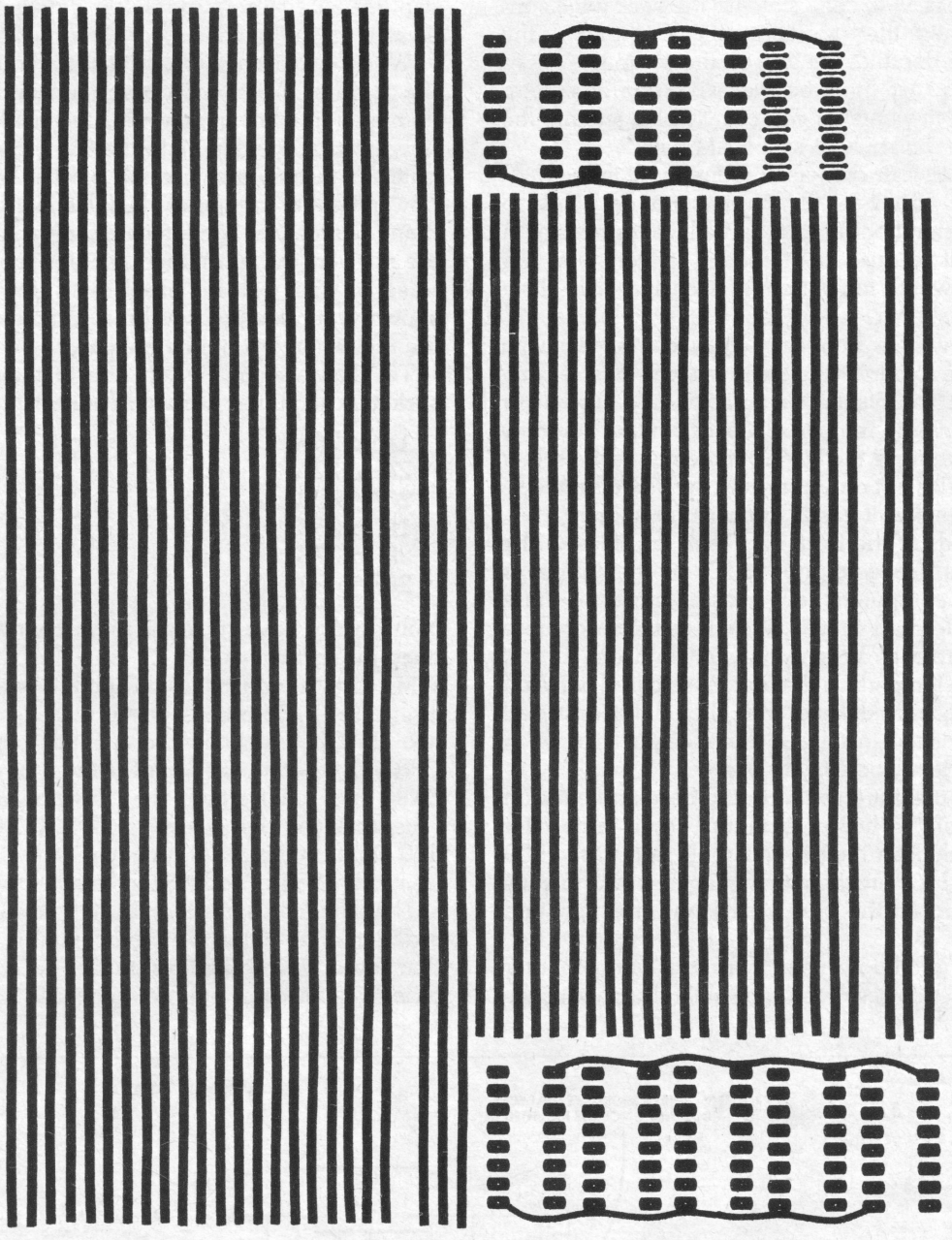
Pin Numbering Conventions

When the board is completed and sitting behind the computer in its normal position, pin 1A is the leftmost pin toward you on each 100-pin connector. Pin 50A is the rightmost pin on each connector toward you. Pin 1B is the leftmost pin on each connector away from you, while pin 50B is the rightmost pin on each connector on the side away from you. Pins are numbered 1-50 from left to right. This also makes the traces on top of the board the A traces and the ones on the bottom the B traces. Just remember B for Bottom. Disregard the numbers on the bottom of the connectors.

Preparing the Board

First we etch the board from the layout shown in layout 1. See box for etching instructions. After etching we are ready to mount our edge connectors. They start 1.5 inches from the front end and are spaced 0.75 inch apart. Use a no. 69 drill to drill holes *between* the traces to reach the bottom side of the board. Try to keep as much in between the traces as possible. You should measure out slot spacing and draw lines with a ruler on the board. Use these lines as a guide to keep all the holes in line.

Now use a 1/8 inch drill to countersink just a little. Do this top and bottom. This will cut any trace away from the holes and make it easier to insert the edge connector pins later.



Layout 1.

Remove pins 4A, 4B, 29A and 29B from the connector. Simply push the pins out the top by pushing up from the bottom.

To mount the edge connectors, bend all the A (front) pins up 90 degrees and a little to the right. Insert the B (back) pins through the drilled holes, bend the pins over their respective traces, and solder them in place. Cut off any excess lead with a pair of "nippy" cutters, available from Radio Shack. Now bend the A pins to line up perfectly with their traces on the top and solder them in place. Do this for each connector you want to install. It is best to install the edge connector furthest from the front first and work forward. This gives you more room to solder the front (top) connections.

Now you need to solder a 23-pin edge connector to the front of the board to allow it to plug into the computer's rear edge connector. Take a 100-pin connector and remove pins 1A, 1B, 25A and 25B. Use a hacksaw to cut the connector at the slots just vacated. Now remove the 2 pins that correspond to the slot in the computer's connector. Fill the slot vacated by these pins with a piece of plastic or etched PC board material. (Do the same for all the connectors mounted on your board.) Solder your new 23-pin connector to the front of the ex-

pansion board by bending the pins inward slightly.

Use your hacksaw to cut a slot into the back edge connector plug, where the trace is missing. Cut it about the same width and depth as the one on the computer's connector plug. This will now be the expansion connector where you plug in whatever you usually plugged into the computer's connector, such as a RAM pack.

Drill holes in all four corners and one more in the middle of each side. Install a 1/2 inch spacer at each hole to raise the expansion board to the same level as the slot in the computer. If you use hex threaded spacers, you can put a small screw into the bottom of each spacer to allow exact level adjustment.

Plate the new rear connector plug completely with solder. Now use solderwick to remove the solder, leaving a thin plating like on Sinclair's connector. Use acetone or nail polish remover to remove all traces of flux. You may also use this to remove the flux from the rest of the solder coated traces. This not only improves the board's appearance, but also makes it easier to check for shorts.

IC socket pads at the front and back left sides of the board give you space to mount ICs in the future as desired. Apple computers have sockets similar to these on their boards for user circuits. We will use a few of these later in these projects. When you want to use a socket, wire the IC with jumpers on the bottom soldered between the pads and traces as desired. You should drill each pad with a no. 69 drill and install a socket in each. Then when you need one, it will be easy to use.

You can use the extra room on the front and back on the right side of the board to mount some large filter capacitors to help filter the power supply. Mine are 1500 mF at 35 volts. Just glue them to the top and drill holes as necessary to connect the leads to the power supply and ground traces.

If you want a good power supply to run your computer and all your other devices, you can solder the power supply wires directly to traces on your expansion board. You can use an available supply, or you can build one from scratch (see schematic 1).

When you use a supply this way, you apply power to the computer from your expansion board, rather than the computer providing power to the board. You will not need to plug anything into the 9V DC jack on the computer. You may also use the Sinclair supply to provide power via the computer, but this limits you as to how many boards you can attach at one time. These devices are not very current-hungry, so you may get away with three or four with no problem. If you use the supply shown, your computer will run completely cool because the regulation will not be onboard.

To allow use of the left side of the edge connector sockets, use jumper wires to connect:

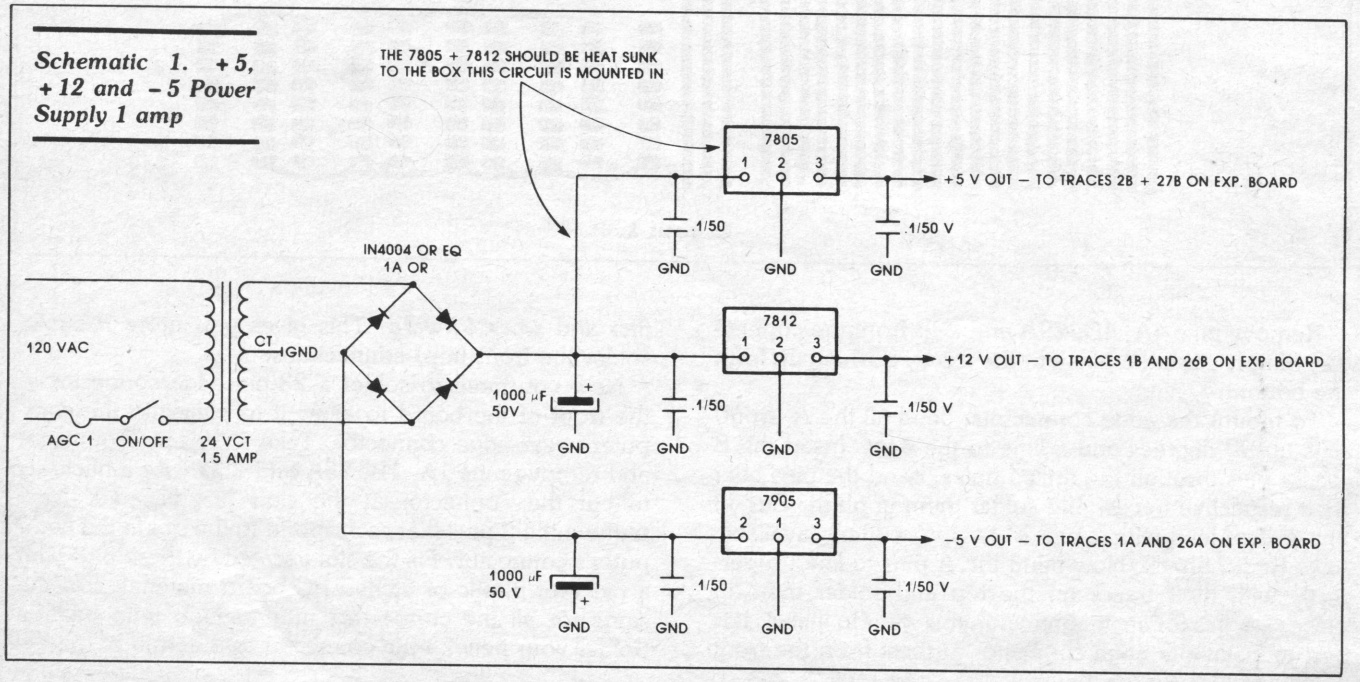
1A to 26A
2A to 27A
25A to 50A
1B to 26B
2B to 27B
25B to 50B etc.

You do not have to make these connections until you need the extra slots.

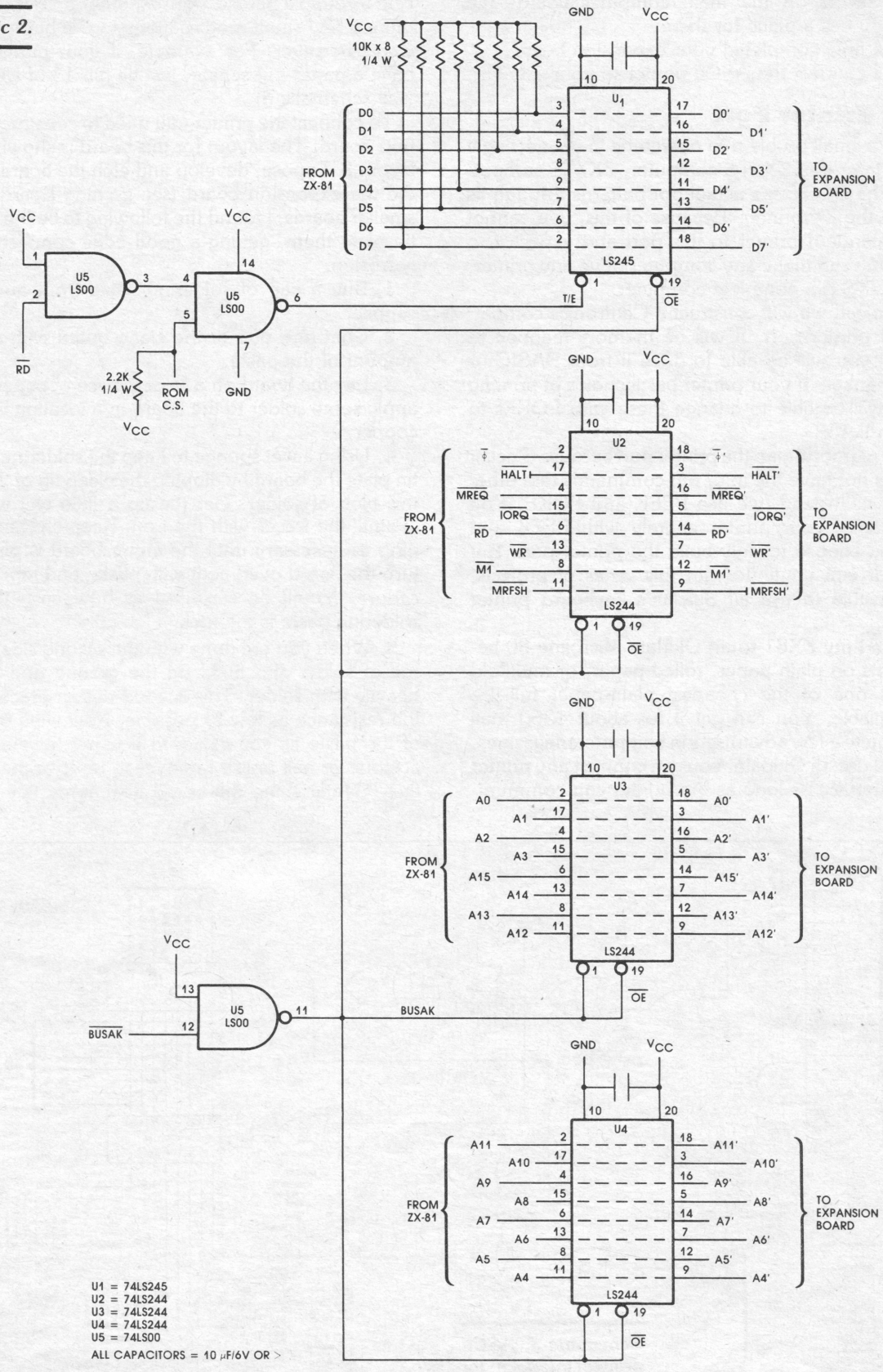
My ZX81 is capable of driving this board, loaded with three 16K RAM boards of my own design (total 48K), two EPROM read boards, one printer port board and the EPROM programmer board without any problems. My ZX80 does not like more than three boards on it at once. I suspect if you change resistors R4-R11 from 1K Ohm to 470 Ohm on the ZX80, you would get as much drive as from the ZX81. The ZX81's resistors are at this value.

The schematic shown for the buffer board (schematic 2) is designed to go between the computer and the expansion board. The design works well. You could assemble it on a small piece of perforated board. You could mount

Schematic 1. +5, +12 and -5 Power Supply 1 amp



Schematic 2.



U1 = 74LS245
U2 = 74LS244
U3 = 74LS244
U4 = 74LS244
U5 = 74LS00

ALL CAPACITORS = 10 μF/6V OR >

BUS BUFFER BOARD
TRI-STATES IF $\overline{BUSAK}$ = ϕ
E.F.S. 8217

FANOUT = 75 STANDARD TTL LOADS HIGH
15 STANDARD TTL LOADS LOW
(BOTH LS244 & LS245)

V_{CC} = +5VDC

the 10K resistors on the main computer board—the ZX81 already has a place for them.

You have now completed your expansion board.

Next we'll connect the ASCII printer to your ZX/TS.

Parallel Printer Port

We have a small problem to overcome in connecting a standard printer to a ZX/TS computer. ZX/TS computers output the character's actual dot patterns through its I/O port to the ZX printer. Because of this, you cannot connect a standard printer to this port and expect it to work. But you can make any computer drive any printer, and the ZX/TS is a complete computer.

In this project, we will construct a Centronics-compatible parallel printer port. It will be memory-mapped at FFFFH and you will be able to drive it from BASIC or machine language. If your printer has a choice of printing fonts, you will be able to change these with POKEs to 65535 from BASIC.

I chose to memory-map the printer port because Sinclair BASIC does not have the in or out commands that other BASICs have. Instead, we use PEEK and POKE. You may even drive your printer entirely with PEEK and POKE if you choose to only build the printer port. But remember, if you continue with this series of projects, you will be able to use all Sinclair's keyboard printer commands.

I connected my ZX81 to an Okidata Microline 80 because it prints on plain paper, rolled paper, or multifold paper. It is one of the cheapest plain-paper full-size printers available. You can get it for about $350 mail order from quite a few advertisers in computer magazines.

Although I use an Okidata, you can connect any printer using this interface as long as the printer can communicate through a parallel port. Another printer type may require a few small modifications to the hardware or software, however. For example, if your printer does not have a paper-out sensor, just tie pin 13 of U6 to ground (see schematic 3).

To connect the printer you need to construct the printer port board. The layout for this board is shown full size as layout 2. Expose, develop and etch the board just as you did the expansion board (see Etching Boards box). On smaller boards, I found the following to be the best way to tin plate them, getting a good edge connection and no corrosion:

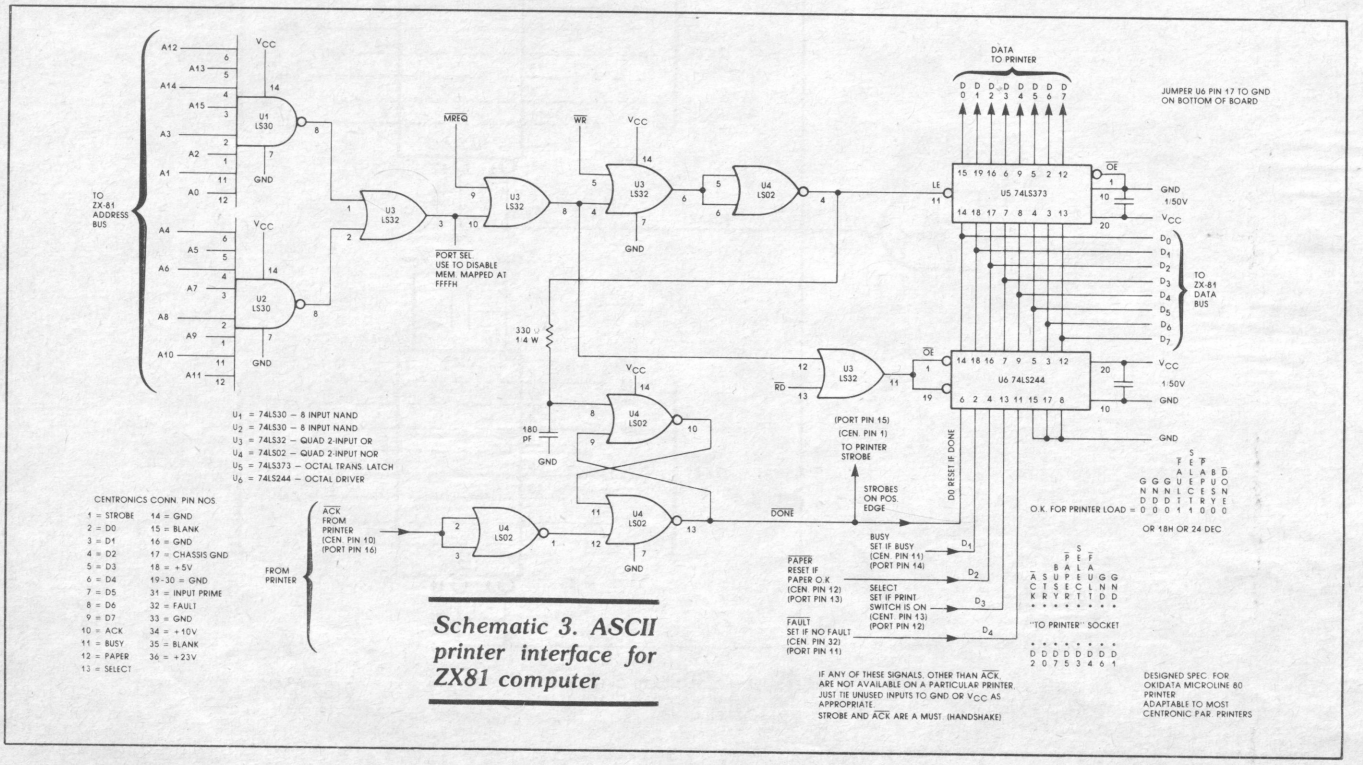
1. Buy a can of soldering paste from an electronics supplier.

2. Coat one side of the clean board with a generous amount of the paste.

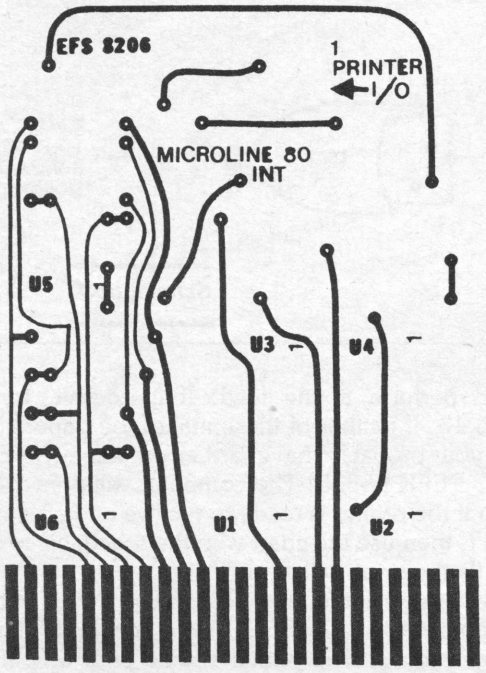
3. Lay the board on a folded piece of paper towel and apply some solder to the board in a location with a lot of copper.

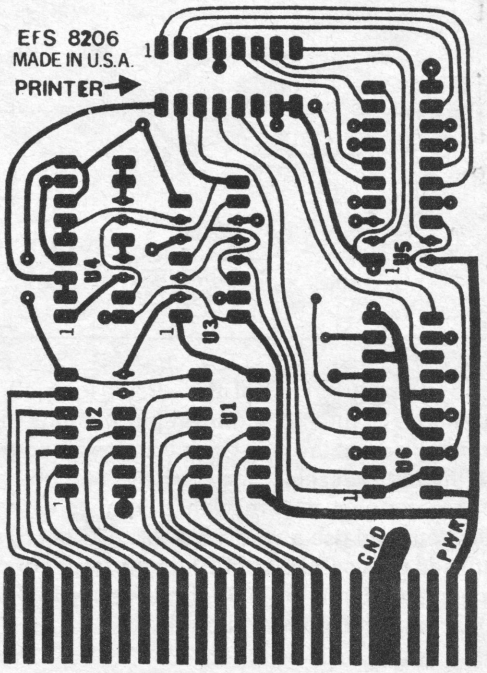
4. Using a wet sponge to keep the soldering iron clean, tin plate the board by dipping the clean tip of the iron into this blob of solder. Get the tip a little wet with solder. "Paint" the traces with the iron. Keep cleaning and dipping as necessary until the entire board is plated. Then turn the board over, coat with paste, and repeat the procedure. You'll be surprised at how easy this is—the soldering paste is the trick.

5. When you are done with the second side, take your roll of solder and build up the ground and Vcc traces heavily with solder. This is good design practice to keep the resistance as low as possible. Now wipe off as much of the paste as you can with a paper towel. Use either acetone or nail polish remover to remove the last of the flux. Gently shine the board with some fine steel wool



Schematic 3. ASCII printer interface for ZX81 computer





Layout 2.

and clean it again with acetone or polish remover. Now take the board to a sink and spray it well with hot water to remove any paste the acetone may have missed.

Your home-made board looks like it came from a factory, doesn't it?

Now drill all the holes from the bottom with a no. 69 drill. After this use a 1/16 inch drill to remove burrs.

The next step is to solder the feedthroughs. This is probably the most time-consuming part of all the projects. Take a wet sponge and lay it out on a baggie. Insert a wire in a feedthrough and solder it to one side. Turn the board over and lay it on the wet sponge to solder the other side. Then clip both sides flush (cut the excess lead as close to the board as possible). "Nippy" cutters from Radio Shack are great for this. The sponge keeps the opposite side from coming unsoldered while you solder the second side.

Do this for all the feedthroughs. Note the four right above U4 are not feedthroughs, so do not solder them. The 330 Ohm resistor goes just above U4, with the 180 pF capacitor just above it.

Now install all the IC sockets, inserting them so they have pin 1 on the side marked on the board (pin 1 down on all but the printer socket, pin 1 to the right on the printer socket). Install a jumper wire from pin 17 U6 to ground and from the feedthrough near pin 2 of U4 to ground.

Clean the flux off the board from the solder and install three or four bypass capacitors (0.1 mF will work fine) under some of the ICs from Vcc to ground. If you have small tantalum caps, they work even better. The board will work without any, but good design practice calls for some.

Use a hacksaw to cut out the contact (fourth from the left) that corresponds to the pin on the edge connector socket. This is the "slot." Try it in the socket for fit and use a small file to shape as necessary.

Plug in the ICs and the port is complete. Be sure to check the board over well for shorts or bad joints before applying power.

The only thing left to do before we test our work is to make an interface cable. On the printer end you will have to use whatever connector the manufacturer used. Use a 16-pin DIP plug for the port end. The Centronics connector used on the Okidata requires an amphenol or equivalent 36-pin male plug (P/N 57-30360). I bought mine from a local distributor for $7.95—it cost more than everything else put together!

Use a piece of 24-conductor ribbon cable (3-4 feet long) to attach the connector to the DIP plug. It is best to have a ground wire between each data line and also the strobe line. This allows the ribbon cable to work as a transmission line. This may not be necessary for only 3-4 feet, but is still a good idea.

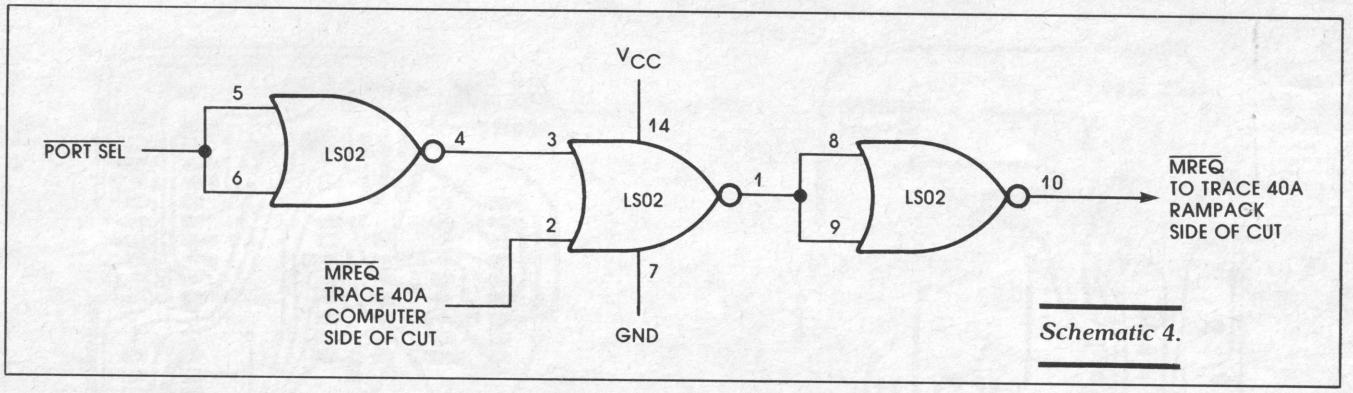
The Centronics pin numbers are marked on the printer. Now simply connect D0 to D0, D3 to D3, STR to STR, etc.

Modification for use with full 48K RAM (Memotech or similar)

If you have a full 48K of RAM, you will need to add the circuit in schematic 4 to the expansion board to keep your RAM mapped at FFFFH from fighting your printer port.

Although your printer may work without this modification (mine does), you really should do it.

Cut trace 40A 3/4 inch from the end of the RAM pack connector (as we did for Sinclair's RAM pack if it was



Schematic 4.

used) and connect pin 10 of this circuit (output) to the RAM pack side of the cut. Connect pin 2 of our circuit to the *computer* side of the cut. Connect pins 5 and 6 of our circuit to the not port select line on the printer board. You may simply connect it here with a jumper wire to the board, or you may use a wire from this (U3 pin 3) to one of the unused traces (50A or 50B) on the edge connector. The circuit can be built on one of our unused IC sockets on the rear of the expansion board.

Decoding RAM

A note here before we go on. For this port to work on the ZX/TS computer, the RAM has to be fully decoded so that no conflicts exist. Follow the instructions on p. 5 of SYNTAX newsletter, May 1982. An even better way to do this for Sinclair's RAM pack would be to throw together the little circuit in schematic 5 on the closest IC socket (the uncommitted ones at the back of the expansion board).

Cut trace 38B about 3/4 inch from the end (remember, the B side is the bottom). Pick up M1 from trace 48A, and A15 from 37B. Feed the output of the circuit (pin 8) to the RAM side of the cut trace 38B. (Pick up A14 from the computer side of the cut.) Don't forget to connect ground and Vcc to your LS00 chip. Your RAM pack is now fully decoded, and you didn't even have to open it up.

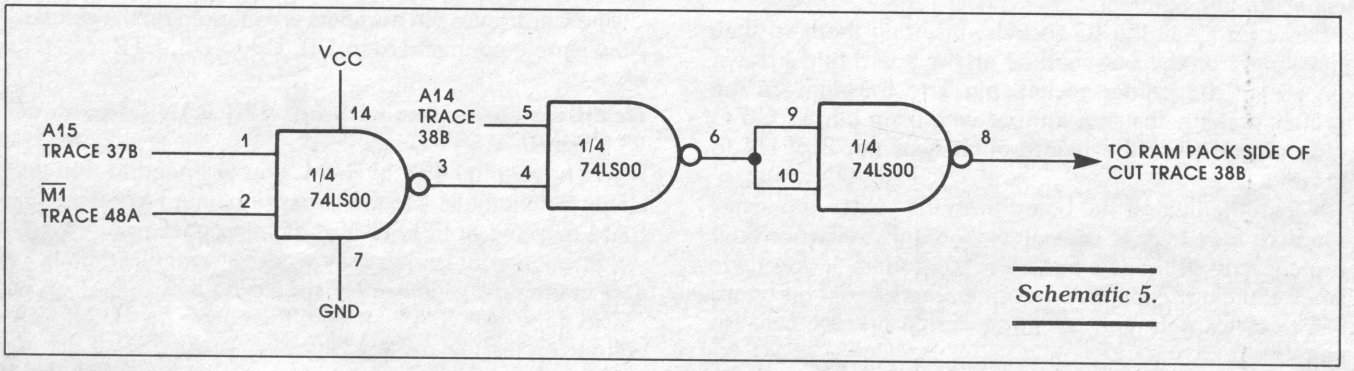
Testing Your Board

Now comes the test. Plug your port board into a socket on the expansion board. Use one on the right side if you haven't wire-jumpered to the left side yet. Connect the cable to the printer and the port board. Turn on both the printer and the computer. Put some paper in the printer. Now enter POKE 65535,10 on your ZX/TS. The printer should perform a line feed. If it doesn't, try POKE 65535,13. If neither of these makes the paper advance, then you probably have an error somewhere. Enter PRINT PEEK 65535. The computer will print 24 on the screen if the printer is ready to receive and all is well. If it doesn't, then use the chart with the schematic to find out why. (Is the print switch on?)

Now let's try some more tests. Type POKE 65535,80. Now enter POKE 65535,13 (80 = ASCII "P" and 13 = carriage return). The printer should print a P. If all this doesn't work, you must go back through your work to find the problem. If you can't find any errors, use an ohmmeter and the schematic to trace continuity from pin to pin as on the schematic. Your PEEK 65535 may also give you a clue as to what is wrong.

We are ready for a more advanced test. Our BASIC driver program inputs a string, converts it to ASCII, and sends it to the printer. This program is only for a printer like the Microline 80 that uses TRS-80 graphic symbols with codes 128 to 255. The conversion codes (in an array in this program) convert as closely as possible the ZX character set up to CHR$ 191. All the characters after this are not converted because they are multi-character "tokens," not actually characters. When we patch into the ROM, later, they will never even reach our driver program. The ROM will have already converted them to characters for us! Our BASIC driver program is listing 1.

Type GOTO 20 and enter the ASCII conversion numbers from the ASCII conversion table. After each entry, a number will be displayed. They should reach powers of 16 at the end of each row. After entering the 64th number, there will be a slight pause. Then the number 129 should appear. Carry on from here with the conversion numbers starting 2080H on the table (listing 9). All the



Schematic 5.

Listing 1.

Here is our BASIC driver program:

```
  10 GOTO 170
  20 DIM A(192)
  30 FOR N=1 TO 64
  35 PRINT AT 8,12;N
  40 INPUT H$
  50 LET A(N)=((CODE H$(1)-28)*1
6)+(CODE H$(2)-28)
  60 NEXT N
  70 FOR N=65 TO 128
  80 LET A(N)=32
  90 NEXT N
 100 LET A(119)=13
 110 FOR N=129 TO 192
 120 PRINT AT 8,12;N
 130 INPUT H$
 140 LET A(N)=((CODE H$(1)-28)*1
6)+(CODE H$(2)-28)
 150 NEXT N
 160 STOP
 170 INPUT H$
 180 FOR N=1 TO LEN H$
 190 IF PEEK 65535<>24 THEN GOTO
 190
 200 POKE 65535,A(CODE H$(N)+1)
 210 NEXT N
 220 IF PEEK 65535<>24 THEN GOTO
 220
 230 POKE 65535,13
 240 GOTO 170
```

20s have been entered for you. After you enter number 192 you will get a report 9/160.

A note here. The Microline printer has a feature that allows it to do both a carriage return and a line feed when only a carriage return is sent. (This is explained in the manual.) If you use this printer, you should move shorting plug S4 on the PC board to side B to use this feature. If you are using another printer, check your manual to see if you have this feature. If your printer has it, then use it. For some printers this is the only way to do a line feed. On others, a line feed (decimal 10) does the same thing. If your printer is this way, then change line 230 to read POKE 65535,10. Also change line 100 to read LET A(119) = 10. In the ASCII convert and copy routine, change the following bytes to 0AH, 1076H, 20EAH, and 2100H.

Now enter GOTO 170. The screen will clear and the computer waits for you to enter a string. Type in whatever you like. You can print any character except the multicharacter tokens or keywords like PRINT and RUN. Inverse characters should print as lower case and the British pound sign "!".

If any characters print incorrectly, you probably made an error while entering the hex data. The first entry is the converted ASCII code for Sinclair code 0. The last entry is converted to ASCII code for Sinclair code 191. The codes are all in order. You can change them as you wish for any necessary conversion. For example, if your printer does not use TRS-80 graphic symbols and codes, then change the codes whose values are higher than 7FH to the necessary codes. You can even use the same hardware and software to convert to other codes if your equipment needs it. Just change the ordered table to be what the equipment needs.

Now that we're tired of just printing strings, let's put the machine language program into memory! This program acts just like the COPY command except we can only use it from within a program. If you try to use it in command mode, you will only accomplish printing 22 blank lines! But from within a program, you may copy the screen to the printer.

The easiest way to get this into memory is either with Sinware's disassembler, or Sinware's "Hot Z." These two programs (especially "Hot Z") are nothing short of fantastic! If you haven't got a copy of "Hot Z" yet, get one as soon as possible. It only cost $19.95 pp, and is worth five times the price.

Lacking one of these programs, you may use any available hex loader. They all perform the same function (see listing 2).

Listing 2.

```
  10 LET X=28672
  20 INPUT A$
  30 POKE X,((CODE A$(1)-28)*16)
+(CODE A$(2)-28)
  40 PRINT X;"=";A$
  50 IF PEEK 16442<3 THEN PAUSE
60
  60 IF PEEK 16442<3 THEN CLS
  70 LET X=X+1
  80 GOTO 20
```

Note that the machine language program will start at location 28672. This is hex location 7000! Enter the code in hex starting with the conversion table (listing 9) and ending with C9 (RET) at the bottom of the listing (listing 10). In effect, the listing in RAM will be exactly as shown in the listing, except all the addresses will actually be 7XXX rather than 2XXX. The only changes to the listing that will have to be made for this change in addresses are the following:

Location listed as 20D3H should be change to 70 (was 20) (now CALL 70D9).

Location listed as 20DEH should be changed to 70 (was 20) (now LD HL, 7000).

Before entering this loader program and the hex code, POKE 16389,112 and then the NEW command. This will set RAMTOP at 28672 (7000H).

If you have one of Sinware's programs, then SAVE the machine code only to tape with its tape SAVE utility. After you have the machine language in protected memory, if you are using one of the Sinware programs, disassemble it to make sure it is correct. Now, if you are using "Hot Z," enter 70C0 in READ mode, hit "W" for WRITE

mode, then "R" to RUN. If you are using the "Sinware Disassembler," from menu ENTER "M" for machine language monitor, 70C0 to point at the correct location, and "R" to RUN it. With either of these, your printer should "COPY" the top 22 lines of the screen!

If you are not using either of these programs, then our test will be a little harder. You can try the little program (listing 3) that will print the Sinclair character set on the screen and then copy it, when it has printed 21 lines. Press any key, and the screen will clear and the program will continue until it reaches the end of the character set.

Listing 3.

```
  10 FOR N=0 TO 255
  20 PRINT CHR$ N;"  ";
  30 IF PEEK 16442=4 THEN GOSUB
100
  40 NEXT N
  50 RAND USR 28864
  60 STOP
 100 RAND USR 28864
 110 PAUSE 4E4
 120 CLS
 130 RETURN
```

The Microline printer's copy is shown in listing 4. Note that the pound sign prints as "!", and the inverse pound sign prints as "#". Inverse letters should also print as lower case.

Listing 4.

```
  " ! $ : ? ( ) > < = +
  - * / ; , . 0 1 2 3
4 5 6 7 8 9 A B C D E
 F G H I J K L M N O P
  Q R S T U V W X Y Z
RND INKEY$ PI ? ? ? ? ?
? ? ? ? ? ? ? ? ? ? ?
 ? ? ? ? ? ? ? ? ? ? ?
  ? ? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ? ? ?
 ? ? ? ? ? ? ? ? ? ? ?
  ? ? ? ? ? ? ? ? ? ? ?
 ? ? ? " # $ : ? ( ) >
< = + - * / ; , . 0 1
 2 3 4 5 6 7 8 9 a b
c d e f g h i j k l m
 n o p q r s t u v w x
  y z " AT TAB ? CODE
VAL LEN SIN COS TAN AS
N ACS ATN LN EXP INT

SQR SGN ABS PEEK USR S
TR$ CHR$ NOT ** OR AND
  <= >= <> THEN TO STEP
  LPRINT LLIST STOP SLOW
 FAST NEW SCROLL CONT D
IM REM FOR GOTO GOSUB
INPUT LOAD LIST LET PAUS
E NEXT POKE PRINT PLOT
 RUN SAVE RAND IF CLS
UNPLOT CLEAR RETURN COPY
```

You can use this machine language program to copy the screen from within any program you wish! The only problem is that you have to have it in memory in order to use it.

If you're like me, you won't want to put this into RAM every time you need it. Our next step then will be to build an EPROM programmer and an EPROM READ board. Then, we can put this program on EPROM, and have it permanently in memory to use whenever we want it.

Next issue: Building your own EPROM programmer and read boards. **SQ**

Etching Printed Circuit Boards

After you gain experience, the process of etching your own circuit boards is fun and easier than developing black and white film. Basically, you start with laminate, a glass- or paper-epoxy substrate coated on each side with copper foil. You clean the surface with steel wool and spray it with a coat of material sensitive to ultraviolet (UV) light. This coating responds to UV light the way film does to visible light.

Then, using a transparent copy of the circuit layout (akin to a photo negative), you shine UV light through the transparency onto the sensitized surface of the board. Next you develop the resulting image. This process leaves a hard, etch-resistant coating wherever you want conductors.

A chemical etchant removes the copper except where the resist pattern (resistant coating) protects it. Remove the resist with acetone or nail polish remover. Then rinse and clean the board again with steel wool. Now you drill the necessary holes and plate the board with solder. Your board is ready for assembly.

You can get materials such as photo resist, developer and etchant from a well supplied electronics house. Pick up other necessities like trays and tongs for handling the boards from photo supply stores.

Here's how to etch the board for your expansion board:

Although visible light should not affect your board, it's better to work with a yellow bug light until after the board is developed.

Layout 1 is the correct size and layout for your expansion board. You may have a transparency made of this, or any of the PC layouts to follow, at most printing stores. A few quick calls around Indianapolis gave charges ranging from $0.60 to $1 each. About half the stores I called provided this service.

This positive transparency is then used to directly expose a positively sensitized PC board. You may use a pre-sensitized board, or you can sensitize one yourself with positive spray-on sensitizer. I use INJECTOR-ALL no. PC197-3 positive spray resist and VECTOR no. 0088-21-45 positive developer.

Apply clear "contact" plastic film on the side opposite the side you're working on. Spray the other side of

the board with resist and let it dry overnight in a dark drawer. Sandwich the board and the transparency under glass—a picture frame works well. Then expose the board through the transparency with a 275 watt "sunlight" bulb for 8 minutes from a distance of 12 inches. Develop the board with diluted developer (dilution directions on bottle) until the pattern is visible and development has stopped. Use a buglight or darklight to see while you work with the undeveloped, sensitized board.

Now rinse the board in water to remove the developer. If the side you just did is to be the top of the board, scrape off the etch resist for the IC pads at both ends. You want these pads only on the bottom of the board.

Next, etch the board. Turn the transparency on and line it up at the edge. You may use either ferric chloride or ammonium persulfate as the etchant. You'll find etching goes much faster if you etch the board face down, with small pieces of plastic in your etching tray to keep the board up off the tray bottom.

When etching is complete, rinse the board off in water and remove the contact film from the back side. Put a new piece of contact film on the side you just etched. Use the same transparency *(for this board only)* for the unetched side turn the transparency over and line it up at the edges. You may use Scotch tape to hold it in place when it is lined up correctly. Repeat the exposure and development steps for this side. You'll use these same steps for all the boards in these projects, with slight variations. After etching, clean the resist off with acetone or nail polish remover, then rinse with water and polish with steel wool.

Now that you've etched the board, you should do something to keep the copper traces from corroding. When the project is complete you can spray the board with clear enamel. Or you can do as I did and just coat all the traces at this point with solder. Using solder improves the conductivity of the traces. This board will have lots of signals going back and forth on it and the solder keeps the impedance of the traces low.

All the power supply traces should get an extra heavy coat of solder. This improves regulation and allows the filter capacitors to do their jobs even better.

Centronics Printer Standard

At SQ, we built most of the circuits in this article and successfully interfaced a ZX/TS computer to a Centronics 779 printer. However, Centronics printer interfaces are not all the same. In fact, some so-called Centronics compatible printers do not even accept the same signals as others. Here is what we found:

The STROBE signal generated by John Oliger's printer interface may not be appropriate for your printer. If the interface does not work but does occasionally print a character or two, piggyback a 7404 (or 74LS04) on U4 of the printer port. Bend up all the pins except 7 and 14 (solder 7 and 14 to the corresponding pins on U4), then bend pin 5 down and solder it (bent diagonally) to pin 4 of U4. Then cut the trace coming from the STROBE pin of the 16-pin interface socket and wire it to pin 6 of the 7404. This provides a more pure negative logic strobe signal. If your printer requires a positive strobe signal, try taking the strobe signal from pin 10 of U4 instead of pin 13. If that doesn't work, try piggybacking the chip as above, but run the signal through two of the inverters instead of just one.

ASCII Convert and Copy Routine

Multilength Character Lines. (POKE system flag 16507 the number of 'ENTERs' to ignore. 16508 is used as a counter.) **(Listing 9.)**

ASCII Conversion Table

2000	20	81	82	83	94	95	96	97
2008	FF	FC	83	22	21	24	3A	3F
2010	28	29	3E	3C	3D	2B	2D	2A
2018	2F	3B	2C	2E	30	31	32	33
2020	34	35	36	37	38	39	41	42
2028	43	44	45	46	47	48	49	4A
2030	4B	4C	4D	4E	4F	50	51	52
2038	53	54	55	56	57	58	59	5A
2040	20	20	20	20	20	20	20	20
2048	20	20	20	20	20	20	20	20
2050	20	20	20	20	20	20	20	20
2058	20	20	20	20	20	20	20	20
2060	20	20	20	20	20	20	20	20
2068	20	20	20	20	20	20	20	20
2070	20	20	20	20	20	20	0D	20
2078	20	20	20	20	20	20	20	20
2080	FF	FE	FD	FC	EB	EA	E9	E8
2088	FF	83	FC	40	23	24	7E	25
2090	7B	7D	3E	3C	26	2B	5F	2A
2098	2F	60	27	2E	30	31	32	33
20A0	34	35	36	37	38	39	61	62
20A8	63	64	65	66	67	68	69	6A
20B0	6B	6C	6D	6E	6F	70	71	72
20B8	73	74	75	76	77	78	79	7A

the board with resist and let it dry overnight in a dark drawer. Sandwich the board and the transparency under glass—a picture frame works well. Then expose the board through the transparency with a 275 watt "sunlight" bulb for 8 minutes from a distance of 12 inches. Develop the board with diluted developer (dilution directions on bottle) until the pattern is visible and development has stopped. Use a buglight or darklight to see while you work with the undeveloped, sensitized board.

Now rinse the board in water to remove the developer. If the side you just did is to be the top of the board, scrape off the etch resist for the IC pads at both ends. You want these pads only on the bottom of the board.

Next, etch the board. Turn the transparency on and line it up at the edge. You may use either ferric chloride or ammonium persulfate as the etchant. You'll find etching goes much faster if you etch the board face down, with small pieces of plastic in your etching tray to keep the board up off the tray bottom.

When etching is complete, rinse the board off in water and remove the contact film from the back side. Put a new piece of contact film on the side you just etched. Use the same transparency *(for this board only)* for the unetched side turn the transparency over and line it up at the edges. You may use Scotch tape to hold it in place when it is lined up correctly. Repeat the exposure and development steps for this side. You'll use these same steps for all the boards in these projects, with slight variations. After etching, clean the resist off with acetone or nail polish remover, then rinse with water and polish with steel wool.

Now that you've etched the board, you should do something to keep the copper traces from corroding. When the project is complete you can spray the board with clear enamel. Or you can do as I did and just coat all the traces at this point with solder. Using solder improves the conductivity of the traces. This board will have lots of signals going back and forth on it and the solder keeps the impedance of the traces low.

All the power supply traces should get an extra heavy coat of solder. This improves regulation and allows the filter capacitors to do their jobs even better.

Centronics Printer Standard

At SQ, we built most of the circuits in this article and successfully interfaced a ZX/TS computer to a Centronics 779 printer. However, Centronics printer interfaces are not all the same. In fact, some so-called Centronics compatible printers do not even accept the same signals as others. Here is what we found:

The STROBE signal generated by John Oliger's printer interface may not be appropriate for your printer. If the interface does not work but does occasionally print a character or two, piggyback a 7404 (or 74LS04) on U4 of the printer port. Bend up all the pins except 7 and 14 (solder 7 and 14 to the corresponding pins on U4), then bend pin 5 down and solder it (bent diagonally) to pin 4 of U4. Then cut the trace coming from the STROBE pin of the 16-pin interface socket and wire it to pin 6 of the 7404. This provides a more pure negative logic strobe signal. If your printer requires a positive strobe signal, try taking the strobe signal from pin 10 of U4 instead of pin 13. If that doesn't work, try piggybacking the chip as above, but run the signal through two of the inverters instead of just one.

ASCII Convert and Copy Routine

Multilength Character Lines. (POKE system flag 16507 the number of 'ENTERs' to ignore. 16508 is used as a counter.) **(Listing 9.)**

ASCII Conversion Table

2000	20	81	82	83	94	95	96	97
2008	FF	FC	83	22	21	24	3A	3F
2010	28	29	3E	3C	3D	2B	2D	2A
2018	2F	3B	2C	2E	30	31	32	33
2020	34	35	36	37	38	39	41	42
2028	43	44	45	46	47	48	49	4A
2030	4B	4C	4D	4E	4F	50	51	52
2038	53	54	55	56	57	58	59	5A
2040	20	20	20	20	20	20	20	20
2048	20	20	20	20	20	20	20	20
2050	20	20	20	20	20	20	20	20
2058	20	20	20	20	20	20	20	20
2060	20	20	20	20	20	20	20	20
2068	20	20	20	20	20	20	20	20
2070	20	20	20	20	20	20	0D	20
2078	20	20	20	20	20	20	20	20
2080	FF	FE	FD	FC	EB	EA	E9	E8
2088	FF	83	FC	40	23	24	7E	25
2090	7B	7D	3E	3C	26	2B	5F	2A
2098	2F	60	27	2E	30	31	32	33
20A0	34	35	36	37	38	39	61	62
20A8	63	64	65	66	67	68	69	6A
20B0	6B	6C	6D	6E	6F	70	71	72
20B8	73	74	75	76	77	78	79	7A

Listing 10.

```
20C0    1616      COPY    LD D,16          ;Set 'D' for 22 lines
20C2    2A0C40            LD HL,(DFILE)    ;Get screen memory address
20C5    23                INC HL           ;Point at first
                                           ;character
20C6    1811              JR CONV          ;Go and copy the
                                           ;screen

Calls here (20C8) from monitor:
20C8    3AFFFF    PTCH    LD A,(PORT)      ;Get printer status
20CB    CB5F              BIT 3,A          ;Is print switch on?
20CD    2807              JR Z RETP        ;Return to monitor
20CF    E5                PUSH HL          ;if not save registers
20D0    D5                PUSH DE          ;used for ZX printer
                                           ;use, later
20D1    CDD920            CALL CONV        ;Call ASCII Conv/Drive
                                           ;Routine
20D4    D1                POP DE           ;Restore registers
20D5    E1                POP HL           ;for ZX printer use
20D6    C3E702    RETP    JP TFAS          ;Jump to 'Temp Fast'
                                           ;for monitor
20D9    7E        CONV    LD A,(HL)        ;Get character code to
                                           ;LPRINT
20DA    23                INC HL           ;Point at next
                                           ;character
20DB    E5                PUSH HL          ;Save location for
                                           ;later
                                           ;table
20DC    210020            LD HL,TABL       ;Point at conversion
                                           ;table
20DF    85                ADD L            ;Add displacement code
20E0    6F                LD L,A           ;Put this in "lo" byte
20E1    3001              JR NC BRK?       ;Jump over inc if no
                                           ;carry
20E3    24                INC H            ;Increment "hi" byte
20E4    CD460F    BRK?    CALL 0F46        ;Check for depressed
                                           ;break key
20E7    3808              JR C CHEK        ;Jump over return if
                                           ;no break
20E9    3E0D              LD A,0D          ;Put ASCII code for
                                           ;carriage return in A
20EB    32FFFF            LD (PORT),A      ;And send it to the
                                           ;printer
20EE    E1                POP HL           ;Clear next address
                                           ;from stack
20EF    CF                RST 08H          ;Return w/report "D"
20F0    0C                REPORT D         ;Break key pressed
20F1    1E02      CHEK    LD E,02          ;Set 'E' to check
                                           ;status twice
20F3    3AFFFF    RDY?    LD A,(PORT)      ;Get printer status
20F6    FE18              CP 18            ;Is it ready to
                                           ;receive?
20F8    20EA              JR NZ BRK?       ;Loop to break test if
                                           ;not
20FA    1D                DEC E            ;Decrement check
                                           ;counter
                                           ;o.k.
20FB    20F6              JR NZ RDY?       ;Loop 'till 2 times
20FD    7E        GTIT    LD A,(HL)        ;Get converted code to
                                           ;send
20FE    FE0D              CP 0D            ;Is it an "ENTER"?
2100    2012              JR NZ SEND       ;Go and send it if not
```

```
2102    217B40    LDN?    LD HL,FLAG      ;Point at multiline
                          ;flag
2105    F5                PUSH AF         ;Save "ENTER" on the
                          ;stack
2106    7E                LD A,(HL)       ;Now, get the flag
2107    23                INC HL          ;Point at the counter
2108    BE                CP (HL)         ;Ready to send "ENTER"
                          ;now?
2109    2805              JR Z CONT       ;Clear counter & send
                          ;enter if true
210B    F1                POP AF          ;Clear the stack
210C    34                INC (HL)        ;Increment counter
210D    15                DEC D           ;Decrement line
                          ;counter
210E    1807              JR 2117         ;Jump over driver,
                          ;this time
2110    F1        CCNT    POP AF          ;Get "ENTER" to send
2111    3600              LD (HL),00      ;Reset the counter
2113    15                DEC D           ;Decrement the line
                          ;counter
2114    32FFFF    SEND    LD (PORT),A     ;Send character to the
                          ;printer
2117    E1                POP HL          ;Get next char address
                          ;to convert
2118    20BF              JR NZ CONV      ;Go do next char if not
                          ;done
211A    C9                RET             ;Return to caller
```


Your ZX/TS Source for supplies —

Maryland Book Exchange

4500 College Avenue
College Park, MD 20704
(301) 927-2510

Not everything is on computer

As useful as your computer is, it can't store all the back-up information and literature you may need. Data Domain has the largest library of computer-related books, magazines and software in the Midwest—and we're happy to point out the titles that you might find useful.

Come to Data Domain and browse—because not everything is on computer.

DATA DOMAIN

1612 E. Algonquin Rd. Schaumburg, IL 60195

– COUPON –
Good for $1 off any book or magazine in stock.
expires Dec. 31, 1982

FIRST TALK TO MINDWARE.

If you're writing software for the Timex-Sinclair personal computer, you ought to talk first to the software publishing company that will do the most for you and your program. Here's six good reasons why good programmers come to us first:

1. MASS MARKETING CAPABILITIES.
The Timex-Sinclair computer market is no longer a mail-order, hobbyist business. We have the marketing resources to put your titles in more than 25,000 retail outlets, including chain stores, mass merchandisers, audio centers and PX's.

2. VERTICAL MARKETING CAPABILITIES.
At the same time, we *also* sell actively to the kind of specialized audiences who can't be reached through mass market outlets. If you've written the definitive program for machine shop estimators, we know where to sell it.

3. BETTER ROYALTY TERMS.
Since we expect to buy the best in Timex-Sinclair software, we expect to pay the best in royalties. And we pay *promptly* and *frequently*—a policy that, sadly enough, many other publishers seem to feel is unnecessary.

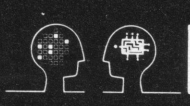
4. "QUICKLOADING".
Every title we sell now incorporates our unique "Quickload" algorithm. "Quickload" enables your program to load and save at a rate six times faster than other Timex-Sinclair software. That kind of special feature means bigger sales—and bigger royalties.

5. PROFESSIONAL PACKAGING AND PROMOTION.
We take our titles seriously. We're willing to invest in good graphics, good documentation, good advertising. You'll be proud to show off your work, because we are too.

6. WE'RE NICE TO DEAL WITH.
Mindware isn't a big, impersonal corporation with layers of bureaucrats to penetrate. We know our leadership in software depends on finding the very best authors—and keeping them. So we try to respond quickly and honestly, and treat you with respect.

We can't promise to publish every program you submit. But if you've got a title you feel is a real winner, our advice is simple:

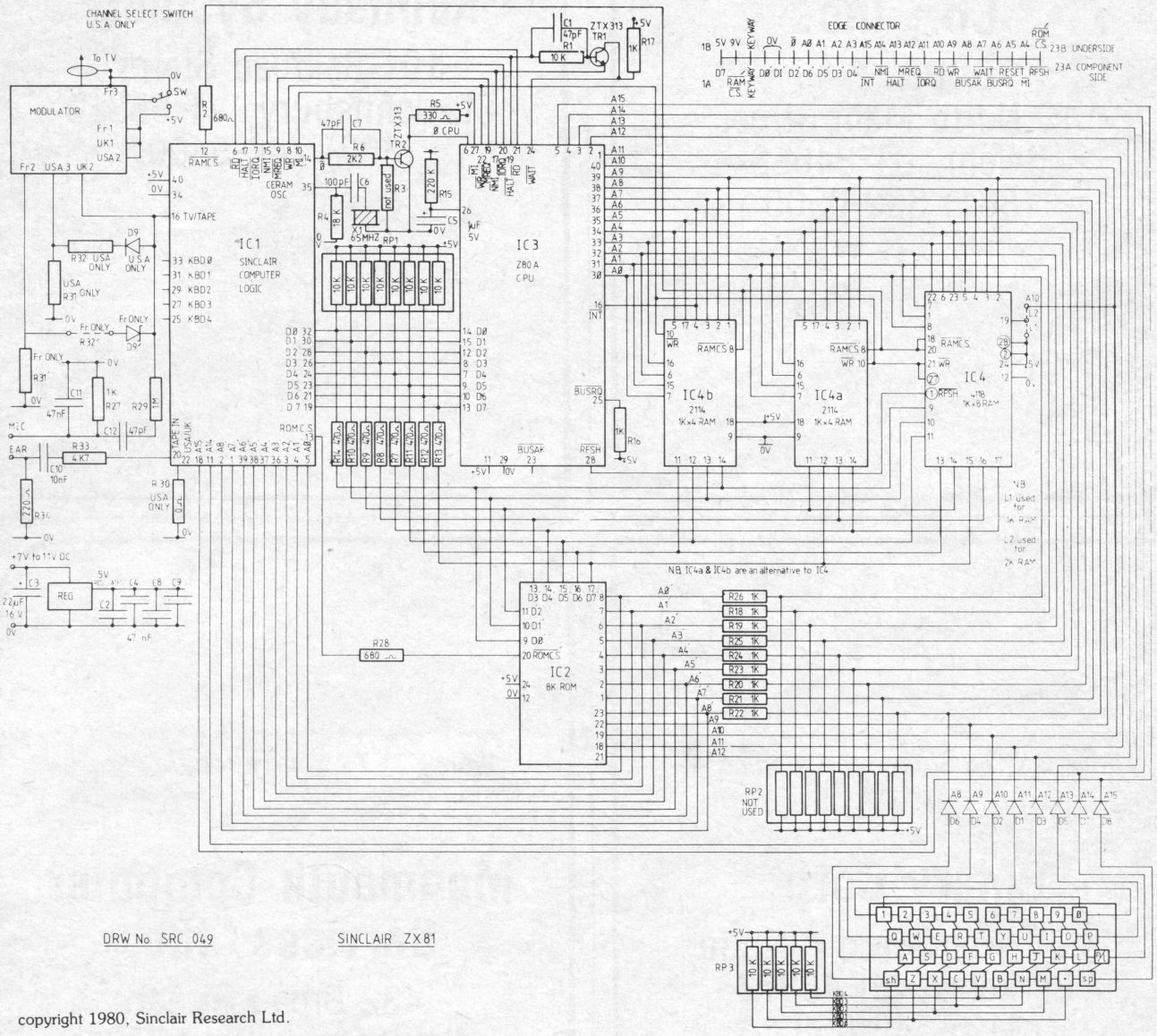
FIRST TALK TO



MINDWARE

15 Tech Circle Natick, Mass. USA 617-655-3388

ZX81 Schematic



copyright 1980, Sinclair Research Ltd.

Your ZX/TS Source for supplies —

The Amusement Game Co., Inc.

Box P
10 North Main Street
Salem, UT 84653
(801) 322-0505

Your ZX/TS Source for supplies —

Kennedy Systems

101 Railroad Street
St. Johnsbury, VT 05819
(802) 748-5400

Your ZX/TS Source for supplies —

Binary Corp.

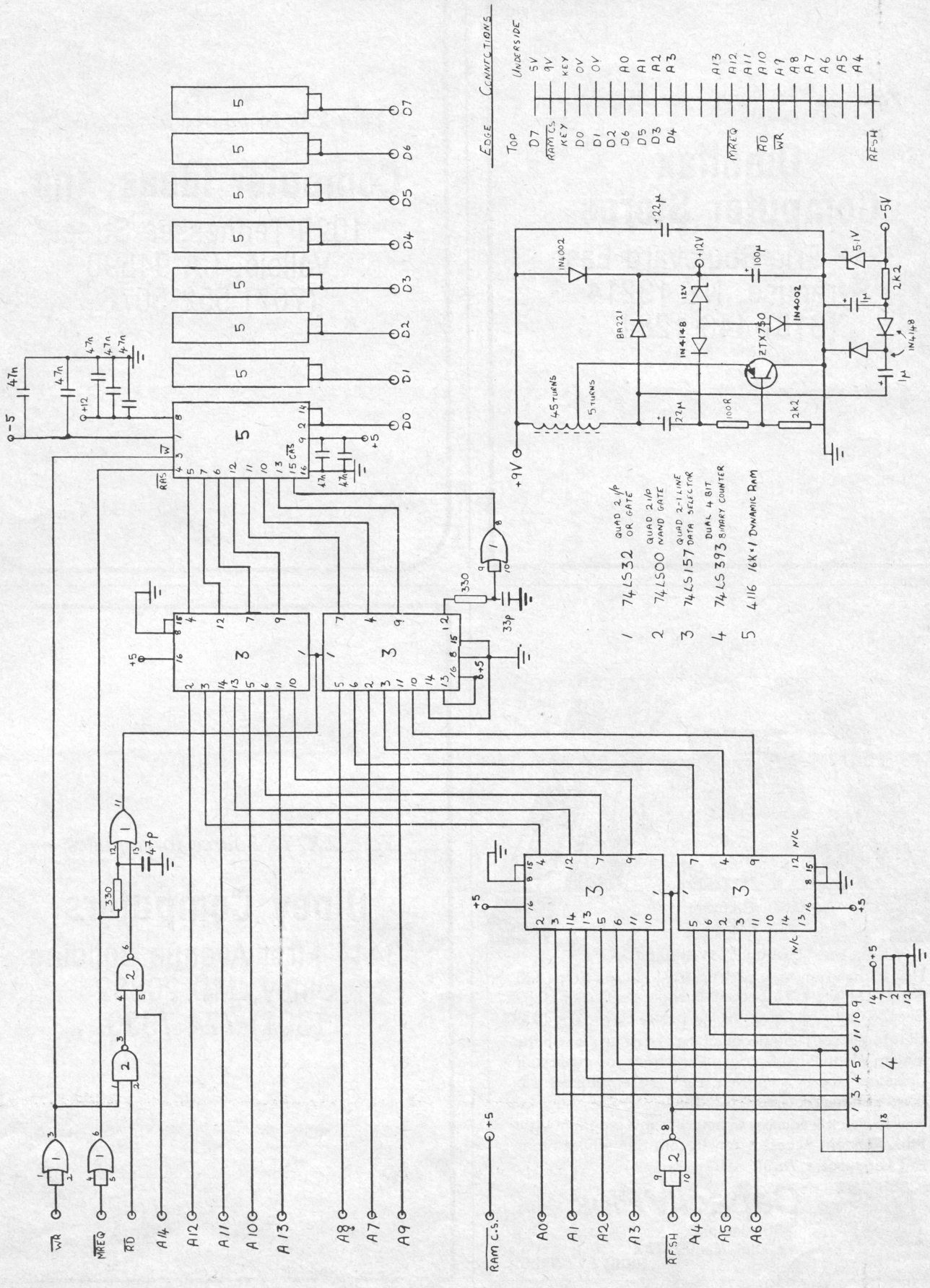
3237 Woodward Avenue
Berkeley, MI 48072
(313) 548-0533

Your ZX/TS Source for supplies —

Monmouth Computer Services, Inc.

432 Broad Street
Shrewsbury, NJ 07701
(201) 747-6745

16K RAM Pack Schematic



copyright 1981, Sinclair Research Ltd.

Your ZX/TS Source for supplies —

Omnifax Computer Stores

3216 Erie Boulevard East
Syracuse, NY 13214
(315) 446-1284

Your ZX/TS Source for supplies —

Computer Ideas, Inc.

1029 Tennessee Street
Vallejo, CA 94590
(707) 552-5076

Software
PLUS

For Any CPM System!

How to make any easy back-up: **Fast Plus** for Xerox 820 Single Density • 17 Second Integrity Verification • 60 Second Copy (directory display before copying) • **75.00**

How to put your knowledge of the system to work in the system: **Menu Plus.** A program which lets you guide your operators step by step even when you're not there. No programming required. • **125.00**

How to check & edit files without writing a program: **View Plus.** Random Access • Any Record Size • Files From Any Language • **75.00**

Capacity Plus
250 Alamaha
Kahului, Maui 96732
(808) 877-3496

Your ZX/TS Source for supplies —

Olney Computers

3414 First Avenue Building
Olney, MD 20832
(301) 774-7330

by Stephen Adams, London, England

ZX Memory Map

When Clive Sinclair designed his basic computer, it was to contain only three things within the memory map (the area storing all the instructions to operate the computer). The area is easily accessible by the Z80A microprocessor and it is easy to program uses for it. The memory area is divided into 65535 locations (64K) by the 16 binary address lines (A0-A15) that indicate the location at which the processor wants to look.

Sinclair required only a place to store users' programs (RAM), a place to hold instructions to operate the computer when it was first switched on (ROM) and a location in memory which would translate the TV picture. So he divided the memory map into three sections: 0-16K for the ROM, 16-32K for the program RAM and 32-64K for the TV picture. He used only the top two address lines A14 and A15 to tell which section was operating at the time.

This divided the memory map into the four sections in Figure 1 with the ROM and RAM repeating themselves again and again. Address line A14 is used to turn on the ROM when the condition is binary 0. When it is binary 1 the RAM is turned on. Address line A15 is used to operate the hardware that puts out the TV picture; it operates only when A15 is binary 1.

Because A14 can also change state from 0 to 1 while A15 is operating the TV hardware, the RAM can be used to store the screen display. The ROM software uses that fact and has only to pick up the address where the screen is— between 16K and 32K— and then add 32K to it to put it on the TV screen.

The only problem with this system is that ROM between 32K and 48K could not be used because it would upset the TV picture. It also meant that the ROM appeared in the 8K to 16K section, blocking out any other uses for that region because it could not be turned off.

That was the situation which existed in the ZX80 and only internal modifications to the computer circuitry could sort it out. The only place left to put new devices produced by other manufacturers was to steal some of the RAM space to fit in the new device. By operating a pin called $\overline{\text{RAM CS}}$ on the expansion board, the RAM could be turned off. Sinclair included this feature in their design because some method was needed to turn off the 1K of internal RAM when the 16K RAM pack was fitted on the back.

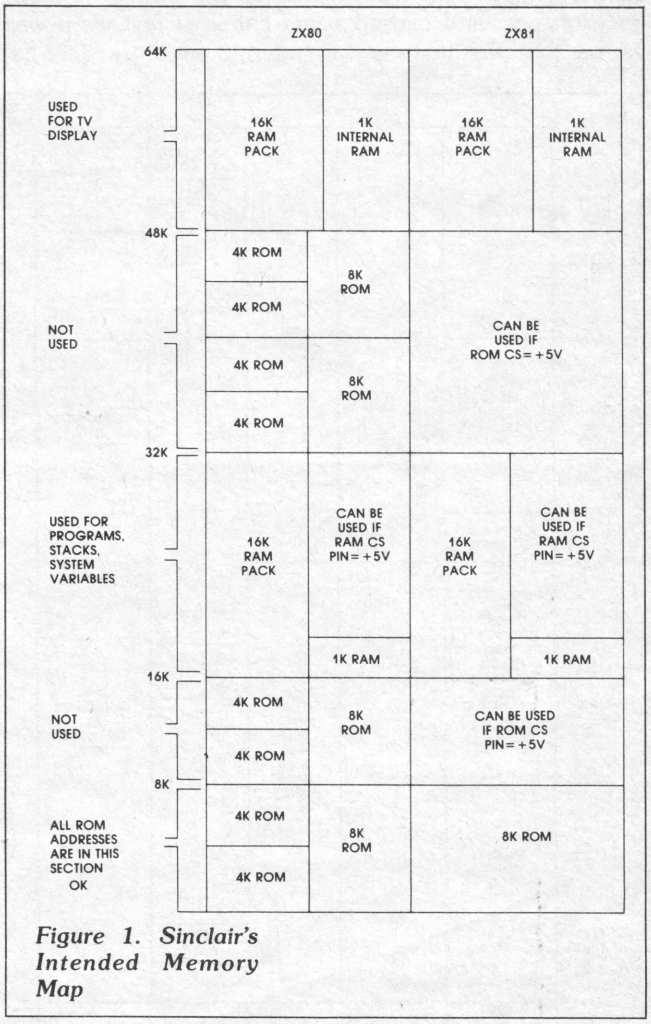


Figure 1. Sinclair's Intended Memory Map

Without turning off the 1K internal RAM, the 16K RAM could not operate. The 1K RAM repeated itself through the entire 16K section and got in the way when 16 separate 1Ks were installed instead. Sinclair's 16K RAM avoided this problem because when attached to the computer it not only blocked off any connection to the expansion port by covering it, but also used the only free memory locations. It also did all of its address decoding inside the pack, so could not be turned off by operating the $\overline{\text{RAM CS}}$ line.

When Sinclair saw the growing market for his computer and that other manufacturers wanted to use it as a basis for experimenting with computers, he redesigned it, giving greater flexibility to the memory map. The result was the ZX81.

The internal design was much the same as the ZX80—Figure 1— but this time an extra connection in the expansion port had been put to good use. It was called the $\overline{\text{ROM CS}}$ pin. By operating it in the same way as the $\overline{\text{RAM CS}}$ pin, the ROM could be taken out of the memory map. This was located on pin 23B on the expansion port and caused some chaos at first, as it was found that the first Sinclair attempt at more external memory, the 3K RAM pack, did not work with the ZX81. This was because inside the pack the pin was permanently set to binary 0, preventing the ROM from being turned on. The only solution to the problem was to cut the connection to the pin within the pack, leaving it unconnected to anything within the pack.

People could then free some space in the memory map for their use from the outside of the computer. That was a great advantage, as many of the people who had bought ZX80s or ZX81s did not want to have to fiddle inside their precious computers with a soldering iron for fear of damaging them.

Now many firms that had cut their teeth in the ZX80 could greatly expand the ZX81's flexibility and use with add-on boards.

This change also led every manufacturer to try to use the same areas for different uses. Also because ZX customers were not willing to pay a great deal for any extras to their machines, costs had to be cut to fit the market. So they took a tip from Sinclair and reduced the number of lines they used for addressing.

But then we had the same problem before—that one port covered more than one memory location; some cover up as much as 8K. That, of course, means that when one device is using that area, no other devices can use the same address.

Figures 2, 3 and 4 show many available add-ons for ZX and TS computers, plus their addresses. These charts also show, by putting into different columns, the devices

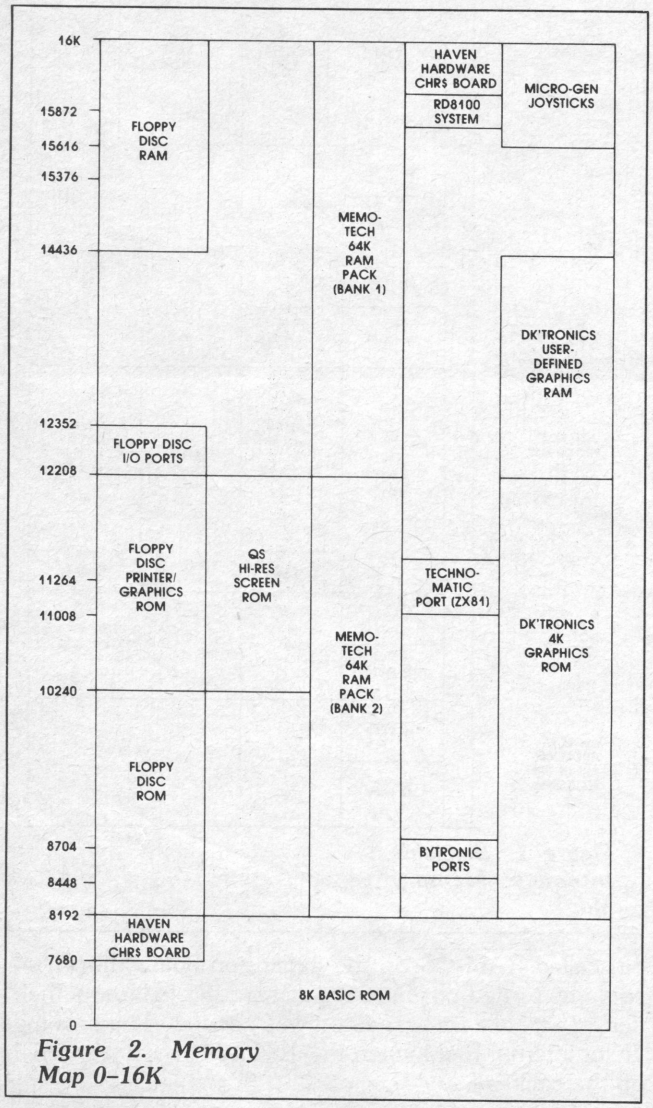


Figure 2. Memory Map 0–16K

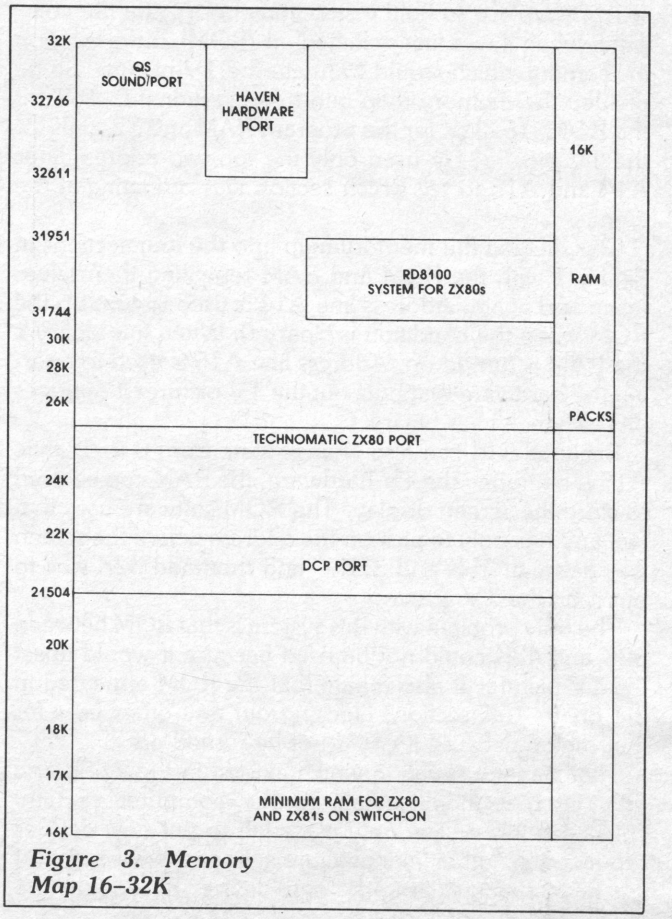


Figure 3. Memory Map 16–32K

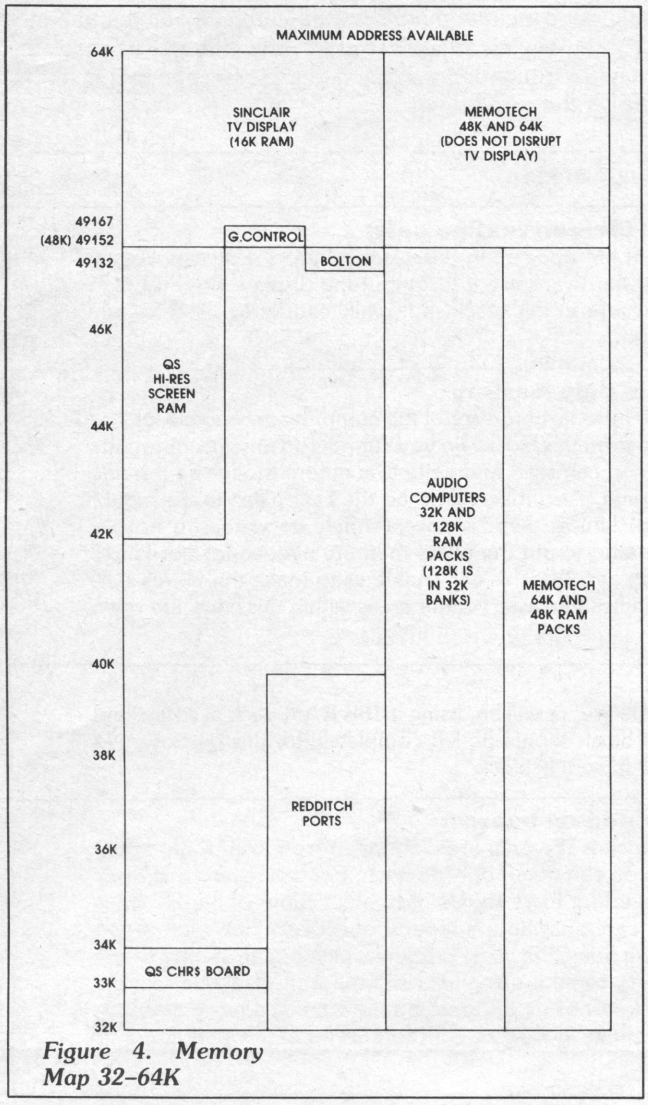


Figure 4. Memory Map 32–64K

which cannot be used together. If I have omitted any devices which fit either the ZX80 or the ZX81, please send me, care of SQ, the name and address of the manufacturer, what the device does and how many addresses it covers, including all the ports if it contains more than one.

I have included two RAM expansions as they all have to start at 16K and work their way upwards. That memory must be continuous for BASIC, as the program would crash if it tried to store memory in a RAM location which was not there.

For more details on these devices, contact the manufacturer (see accompanying list). **SQ**

*In addition to the British companies in Steve Adam's **ZX Memory Map**, these U.S. suppliers have provided SQ with the following memory data.*

Company	Device	Addresses Used
Byte-Back Co.	BB1 control module	32764-32767
	Modem-RS232	38 & 39 in ROM
	M64 memory	8-16K deselected in 2K increments
Memotech	Hi-res graphics board	8-10K
	Centronics parallel interface	10-10.5K
	RS232 interface	10.5-11K
	Balance of addresses open so far, but disk drive I/F *may* run 11-13 or 14K region	
CAI Instruments Inc.	C-40 printer	8-10K plus 2K paging from 14-16K
	Stringy floppy	10-12K
	Modem/ASCII converter	13-14K
Data-assette	ZX99 tape controller	ZX/TS1000's own ROM in memory address space (i.e., from 8192 to 10239) but since the address is not fully decoded, ZX99 will respond to any address in range 8192-16383

EZRA GROUP II
EZRA GROUP II

LOW LOW LOW LOW LOW LOW LOW LOW LOW
PRICES!
Biorhythms 8K ROM/1K&up.....1.00
Graphics Billboard 8/lup.....1.00
Horse Race 8/lup.....1.00
SPINNER TM(like Rubik's)8/16...2.00
Improved SLOW PAUSE.....1.00
Linear Regression 8/lup.....2.00
CHAR. Generator Demos 8/lup....2.95
SPLIT-A-STACK TM SLOW/lup.....2.00
CHEWTER TM(like PACMN)SLOW/lup.2.95
ISLAND SQUARES TM SLOW/2up.....2.00
Random MUSIC!SLOW/lup.....2.00
BASIC Keyword Demos 8/l....from .50
ORDER,SASE,reSASE gets you GOO-
DIES Catalog and FREE! Program

EZRA GROUP II
EZRA GROUP II

POB 5222 San Diego,California 92105

New peripherals for ZX/TS computers have proliferated rapidly since the computer's introduction. Not all peripherals, or add-ons, are compatible with each other, as Steve Adam's accompanying article describes. Because of this problem, Nick Lambert of Quicksilva in England proposed this memory map for future peripherals. If designers agree to this sort of convention, you may be able to use all your ZX/TS accessories at the same time!

ZX Computer Memory Map

64K		
	Screen Display Area	**User RAM (Screen routine only)** The user RAM appears in this block where the screen display routines and hardware use it to output the display. D_FILE can appear anywhere in this block so it really cannot be used for any other purpose.
48K		
	More I/O & Memory Blocks	**Read/write only memory** Due to the internal hardware of the computer no opcode fetches can be made from this block. So you cannot run any machine code from here. You can read and write to memory though so it is fine for data storage. Also it seems to be the best place to do Input/Output from. Unless there is an extremely good reason not to. Quicksilva intend to put the 6K of memory needed for our Hi-res board starting at 32K. This will enable us to make the Hi-res software run significantly faster. Other areas within this block are open to debate.
	Software Switches	
	I/O Space	
	QUICKSILVA Hi-res memory	
32K		
	User RAM	**User RAM** Most people are, or will be, using a 16K RAM pack of some kind so really this block should be left completely for this purpose. No ROM or I/O from this block.
16K		
	?	**ROMs (for add-on boards)** This 8K block is the only area left, apart from user RAM space from which you can do an opcode fetch. For this reason this block should be saved for extra ROMs. Actual functions of the 2K areas in this block can be swapped around of course. However, if two companies are doing the same product which requires some of this ROM space, to avoid wasting valuable space, it would be sensible for both to use the same 2K area. So here it would be advisable to settle for a definite layout, i.e., 8K to 10K for Disc Operating Systems, etc.
14K		
	Voice	
12K		
	Graphics & Extended BASIC	
10K		
	D.O.S. & Network	
8K		
	BASIC	**ROM (Sinclair BASIC ROM)**
0K		

List of Suppliers and Addresses:

Audio Computers
87 Bournemouth Park Rd.
Southand on the Sea, Essex
England

Byte-Back Co.
Rt. 3 Box 147
Brodie Road
Leesville, SC 29070
(803) 532-5812

CAI Instruments Inc.
2559 Arbutus Court
Midland, MI 48640
(517) 687-7343

DCP Port
Microdevelopments
2 Station Close
Lingwood, Norwich
MR 12 4AX, England

Bolton Electronics
44 Newlands Dr.
Bolton, Lancs BL5
BL5 1DP, England

Bytronic
88 Russell Bank Rd.
Sutton Cold Field
W. Midlands C74 4RJ
England

Data-assette
52 South 3d Street
Oxford, PA 19363
(215) 932-3626

Dk'tronics
23 Sussex Road,
Gorleston
Great Yarmouth, Norfolk
England

Ground Control
Alfreda Ave.
Hullbridge, Essex
SS5 6LT
England

Memotech Ltd.
3 Collins Street
Oxford OX4 1XL
Tel. 722102/3/4/5
England

Micro-Gen
24 Agar Crescent
Bracknell, Berks.
England

Redditch
21 Ferney Hill Ave.
Redditch, Worcs.
England

Haven Hardware
4 Asby Rd.
Asby Workington
Cumbria, CA14 4RR
England

Memotech
7550 West Yale Ave.
Suite 200
Denver, CO 80227
(303) 986-1516

Quicksilva
95 Upper Brownhill
Maybush, Suton, Hants.
England

Technomatic
17 Burnley Rd.
London, NW 10
England

CHD PRODUCTIONS

Offers hardware, software and servicing.
Call or write today for information:

CHD Productions
9528 Elvis Lane
Lanham, MD 20706
(301) 577-6645

COMPUTE! Magazine, the foremost resource publication for owners and users of Atari®, PET/CBM™, and VIC-20™ computers, has expanded its coverage to include other personal computers.

We are actively seeking authors to contribute applications articles, utilities, and other software for the Sinclair/Timex computer.

To submit a manuscript, or for more information, write:

Sinclair Editor
COMPUTE! Magazine
P.O. Box 5406
Greensboro, NC 27403

GENERAL SYSTEMS CONSULTING
2312 Rolling Rock Drive
Conley, Georgia 30027
SINCLAIR ZX81 and
TIMEX SINCLAIR 1000 SOFTWARE
16K minimum configuration
Designed to help monitor your finances.

1. Amortizations	9.95
2. Bar Charts	9.95
3. Annuity Evaluation	9.95
4. File Manager	9.95
5. Bank Statement Balancer	9.95
6. Checkbook Simulator	9.95
7. Depreciation Straight Line	9.95
8. Depreciation Declining Balance	9.95
9. Depreciation (ACRS)	9.95
10. Diet Plan	9.95
11. Home Budget	9.95
12. Home Inventory	9.95
13. Home Payables	9.95
14. Home Equity Evaluation	9.95
15. Real Estate Investing	9.95
16. Savings/Investments Analysis	9.95
17. IRS 1040 (Long Form)	9.95
18. IRS 1040A (Short Form)	9.95
19. Income Tax Projections	9.95

Circle selections and fill out form below:
1 2 3 4 5 6 7 8 9 10
11 12 13 14 15 16 17 18 19

Number of items selected ____ @ 9.95 ..... _____
Postage/Handling 1.50 ..... _____
Total: _____

NAME _____
ADDRESS _____
CITY/STATE/ZIP _____

NEW!!!

FROM INTERCOMPUTER, INC.

A full line of software in the areas of business, education, & entertainment, for your Sinclair ZX81 or Timex 1000.

Please send me a FREE program and catalogue.
Name.....
Address

Intercomputer: Inc.
P.O. Box 90
Prudential Center
Boston, MA 02199

SYNTAX

SERVING TIMEX-SINCLAIR PERSONAL COMPUTERS

A PUBLICATION OF **THE HARVARD GROUP**

ISSN 0273-2696

SYNTAX is a monthly newsletter exclusively for ZX80/81/TS1000 owners. We bring you news, reviews and applications for your computer, plus technical notes for circuit-builders. SYNTAX also provides a forum for thousands of users to share advice and problems about programs and vendors. We bring you timely updates about new hardware, software and books. And we cover *all* the Sinclair-technology computers, including the new TS1000.

At SYNTAX we emphasize practicality. You can apply our suggestions even if you aren't sure at first why they work, because we give you complete instructions. Text is clear and easy to understand. SYNTAX readers already know about:

- An automatic phone-dialer they can put together in a few hours
- Syntactic Sums™ to check input for errors
- Programs to explore computer memory
- How to build external additional RAM
- How to add an 8212 I/O chip to control external devices from their computers

And SYNTAX readers like what they get every month. Subscribers know they can depend on us.

After receiving only three issues of SYNTAX, I find that I anxiously await the next. . . keep up the good work!

Martin Irons
Goshen, NY

Congratulations on the brass-tacks, down-to-earth approach of your newsletter. I'll be looking forward to future issues.

Otis Imboden
Washington, DC

Many readers get their first issue and immediately order the back issues—more proof that they like what they see.

You can see what's special about our publication. We work hard to bring you a quality newsletter. We strive to print useful programs of above-average accuracy. As any computer magazine editor can tell you, program listing accuracy is tough to achieve, but we boost our average with every issue. We test each program to make sure it works, it fits in the designated RAM, and it runs when you follow the directions. We print program listings in screen-image format to make it easier for you (it's sure not easier for us!) to enter program accurately. We invented Syntactic Sum™ as an additional aid for you in getting error-free programs. With your subscription you also get access to thousands of other readers, and our staff experts are available by phone to answer your questions or help you solve problems with your machine.

SYNTAX readers get every month:

- Latest news of Z80 hardware and software
- Programs to organize information, calculate, entertain, or instruct
- Do-it-yourself additions
- Clear explanations for beginners

To share the benefits of SYNTAX, just indicate your choices on the order coupon and return it with your choice of payment in U.S. funds. (Please note that additional postage is required for delivery outside North America.)

We are so sure you'll find SYNTAX useful that we promise to refund your entire subscription fee if you aren't satisfied. An unconditional guarantee—you can't lose.

Join the others who stretch the ZX/TS to the utmost. Act now—as soon as we receive your coupon with payment, your first issue will be on its way. For faster service, phone your credit card order to 617/456-3661. Don't miss SYNTAX!

Fill out the coupon below and mail it to: SYNTAX/**SQ**, RD2 Box 457, Harvard, MA 01451

SQ1182

☐ My check is enclosed.
Make checks payable to:
SYNTAX ZX80, Inc.

☐ Please charge my ☐ VISA
☐ Diner's Club ☐ American Express ☐ Mastercard
☐ Carte Blanche account.

Account number _____

Exp. date _____ Bank number (MC only) _____

Signature _____

Name _____

Address _____

City _____ State _____ Zip _____

Day phone ( ) _____ Evening Phone ( ) _____

YES! Please send me:

☐ The Combination (12 issues of SYNTAX and 5 issues of SQ) $39
☐ The Catch-up (SYNTAX Jan. 82-Dec. 83, 5 issues of SQ, plus 1 binder) $77
☐ **The Works** (SYNTAX Nov. 80-Nov. 83, SQ Nov. 82-Nov. 83, plus 2 binders, a 50% savings) $97
☐ 12 issues of SYNTAX $29
☐ 4 issues of SQ, The Syntax Quarterly $15
☐ 1 Premier Issue of SQ $4.95
☐ 1 Binder . . . . . $9 ☐ 1 issue of SYNTAX $4

These offers expire 31 Jan. 83—SUBSCRIBE NOW.

Increase Your Computer's Utility With Our Help

You own a powerful computer, capable of sorting, analyzing, calculating, displaying and manipulating data as well as measuring and controlling. Despite its small size, your computer will help you learn, analyze your business, keep your records or your schedule, dial your phone, send messages, or talk to other computers. To control the power available to you, you'll need hardware and software to use with your machine. Although you can buy many of these products—and we'll tell you about them—some you must create or modify for yourself. SQ will provide you with the complete information to let you use your programming or tinkering time efficiently.

Our experts also bring you SYNTAX, the newsletter, for up-to-date, concise information. You need news and new product announcements quickly. And SYNTAX packs a lot of information into its brief, time-saving format.

But we recognized your need for material too long to fit in SYNTAX. Syntax Quarterly fulfills that special need. In-depth, extensive, detailed information—that's what SQ's all about. Long programs, big construction articles, and detailed directions help you use your computer to its fullest while learning.

SQ gives you tested, accurate information that you can depend on. We test, we build, we check everything that goes into SQ. Our staff experts work for you; it's as though your full-time staff prepared a report for you every quarter.

At SQ, we specialize in your machine—Timex-Sinclair technology. Our programs and projects work, and, because we test each one before we publish it, they work on *your* machine. All the information in SQ can work for *you*.

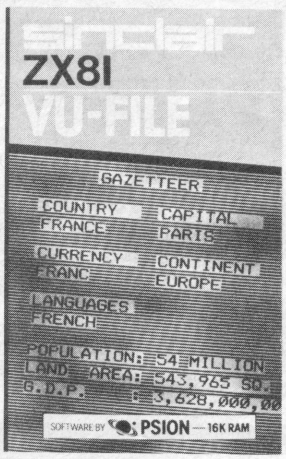
But, to capture these benefits—free software, free plans for hardware, free instruction—you must subscribe. Use the coupon to tell us how to help you best. Or call; we understand people in a hurry. **SQ**

```
 100 LET A$="
 200 LET B$="
 250 LET C$="
 300 PRINT A$,A$
 400 PRINT B$,B$
 500 PRINT B$( TO 7),B$
 600 PRINT B$( TO 7),B$
 700 PRINT A$,B$
 800 PRINT "  ";B$(2 TO ),B$
 900 PRINT "  ";B$(2 TO ),B$
1000 PRINT B$,B$
1100 PRINT A$,A$
1200 PRINT AT 7,22;C$
1300 PRINT AT 9,24;C$
1400 FOR I=0 TO 248
1450 IF I>66 AND I<128 THEN GOTO
 1600
1500 PRINT CHR$ I;
1600 NEXT I
1700 PRINT AT 10,0;"  WE SPEAK Y
OUR LANGUAGE
SYNTACTIC SUM:  22174, 8K ROM
```

```
  WE SPEAK YOUR LANGUAGE     0123
456789ABCDEFGHIJKLMNOPQRSTUVWXYZ
RNDINKEY$PI
 SIN COS TAN ASN ACS ATN LN EXP
INT SQR SGN ABS PEEK USR STR$ CH
R$ NOT ** OR AND <=>=<> THEN TO
STEP LPRINT LLIST STOP SLOW FAST
 NEW SCROLL CONT DIM REM FOR GOT
O GOSUB INPUT LOAD LIST LET PAUS
E NEXT POKE PRINT PLOT RUN SAVE
```

617/456-3661

617/456-3661



Personal Data Bank

Name: VU-FILE
Type: Business
ROM/RAM required: 8K/16K
Written in: Machine Language and BASIC
Listable?: Yes
Easy to Load?: Yes
Printed listing?: No
Display?: Very good
From: Psion Software
Sinclair Research Ltd.
3 Sinclair Plaza
Nashua, NH 03061

VU-FILE is a copyrighted 16K program developed by Psion Software which allows storage and retrieval with ZX/TS computers. The system arrives on cassette and includes a fan-fold set of instructions which also serve as the label for the cassette box and fit neatly within the box. The instructions are brief, to the point and adequate.

The program is menu-driven. Its main menu allows you to set parameters, reenter the file, save on cassette, or set printer parameters so that the data may be printed sequentially on a printer.

Both the set up of parameters and entering of records are very easy. Once the file is entered and records have been introduced the user has the option of stepping through the records sequentially and automatically, stepping through manually either forwards or backwards, ordering the file by any one of multiple fields, printing the current file on a printer, altering a record, or entering a new record. Ordering and altering functions are very simple to operate. The program will automatically list all files sequentially ordered by the desired character or character string in the desired data field or it will list only the file entries containing a specific character string.

One program deficiency is that you cannot immediately recall a specific file without going to the search mode and entering a character string contained only in that file (this may not be a handicap since most files would be numbered sequentially and you need only go to the search mode and enter the file entry number). The program is easy to load and automatically saves on cassette after you enter an appropriate file name. When saved on cassette, the original program and the data are saved.

Although the program is very useful, it's rather difficult to modify. One major advantage of the program is that it packs data. I was able to enter a file of 35 mm slides which included five fields: number (3 digits), date (up to 7 digits), location (up to 10 characters), and a two line description containing up to 72 characters; this allowed space for 400 slides. One option at the head of the menu is "inform" which immediately prints out the number of entries, the number of fields per entry, and the current capacity in the data file you are using.

Comparison of VU-FILE and MULTIFILE

After comparing VU-FILE with MULTIFILE (reviewed Sept. 82, SYNTAX newsletter) and testing it through several applications, I found several similarities and three key differences.

Both systems allow easy searching of files, stepping through files, saving to cassette and printing with the ZX/TS printer.

Contrasting features include: using a test file of five data fields per entry with a total of 90 characters. MULTIFILE would hold 113 files regardless of the number of characters actually used in each allowed space but VU-FILE would allow from a minimum of 100 up to as many as 400 entries depending on whether or not each data file was full since it packs data. MULTIFILE contains two options not available on VU-FILE: numerical summing and security lock feature.

Which to buy? If you need to sum numerical files or require a security lock then you need MULTIFILE. Otherwise, I think that VU-FILE is more versatile. The pack feature allows more files per program, the cost is approximately half the cost of MULTIFILE, and the printer format option allows data files to be presented without labels for each file or with different labels from the original file storage. This feature would allow mailing lists to be printed without labels.

In summary, VU-FILE is a well planned and produced system, adequately documented, very easy to use, and easy to load and save. It's a good tool to help you with membership lists, slide or record collections and other general data lists which need search and retrieval.

J. Ross Shuping,
Greenville, NC

Checkmate

Name: ZXCHESS
Type: Game
Written in: Machine Language
ROM/RAM required: 8K/16K
Listable?: No
Easy to load?: OK
Printed Listing?: No
Display: Very good
From: Gladstone Electronic Distributors
901 Fuhrmann Blvd.
Buffalo, New York 14203
Price: $24.95

I anxiously waited eight weeks for my ZXCHESS from Gladstone to arrive. (Gladstone did notify me that the program was back ordered.)

The enclosed four-page instruction folder gives details on loading and running the program, as well as how to make entries, choose sides, and

even change sides in mid-game.

Fortunately the cassette has the program recorded twice, because the first recording never loaded though the second seems to be reliable enough.

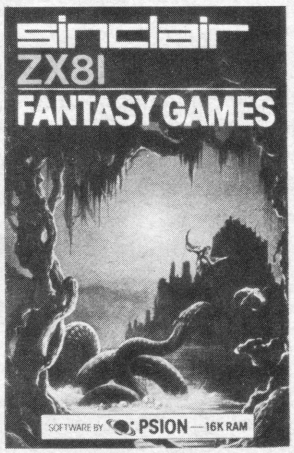
The chessmen are represented by letters on the chess board. An interesting note: The bottom two lines of display normally used for input are apparently used for display of the chess board. But since the program is not listable it would be difficult to find out how that was done.

The only way I could get the program to stop was to unplug the machine. It also cannot be saved on another tape, so be careful with the original.

There are seven levels of play (0 to 6). Level 0 takes about two seconds to reply. Level 4 takes about four and a half minutes. However, I've yet to beat it at even level 0!

I am impressed by the quality of the program in all respects and am happy to find a program that can show off some of the potential of the ZX/TS.

Michael Roberts,
Des Moines, IA



Fantasy Island

Program: Fantasy Games: Perilous Swamp, Sorcerer's Island
Type: Adventure game
ROM/RAM required?: 8K/16K
Written in: BASIC
Listable?: Yes
Printed listings?: No
Easy to load?: OK
Display?: Good
From: Psion Ltd.
Sinclair Research Ltd.
5 Sinclair Plaza
Nashua, NH 03061
Price: $8.95 ($1.95 P&H)

Perilous Swamp is a version of the popular Dungeons and Dragons. In this adventure, your mission is to rescue the princess, who has been captured by the evil wizard, and fight off strange swamp creatures trying to sabotage your mission.

After loading, you are represented on a map by an X. The princess is represented by an asterisk. You can move in any direction—N, S, E, W or any combination thereof.

On your way you can FIGHT, BRIBE or RUN. You start with a certain number of combat points that indicate your strength, and by fighting off the various uglies, you stand to win many treasures—which later give you bribing power when your combat strength ebbs.

The challenge of this game is initially good, but soon becomes predictable. However, it does move quickly and some of the prompts are quite amusing.

Sorcerer's Island is also an adventure game that has you fighting off all kinds of wicked life forms so you can gain their treasure and flee this no-man's land.

This game is almost identical to Perilous Swamp, with a few minor exceptions. Instead of combat points, your strength and endurance are measured in life points. This game also provides you with a map, indicating the topical regions of the island which are either clear, rough, marsh or bush areas. However, you are not represented on this map and it was difficult to tell if any traveling ground was being gained. This game also has more commands: GIVE, FIGHT, REST, MOVE, BRIBE, TAKE, ENTER, MAP, OUT, and INFO. Yet, Island runs slower than Swamp and it, too, soon became very predictable. The prompts are not as entertaining as Swamp.

I've had a difficult time deciding whether to recommend Fantasy Games. Playing Perilous Swamp was fun at first, but the challenge soon wore off. And Sorcerer's Island was so similar and ran so slowly that I could hardly keep from yawning. Yet, if you are the kind of computerist who craves any kind of adventure your video screen can offer, then both these programs may very well strike your creative fancy.

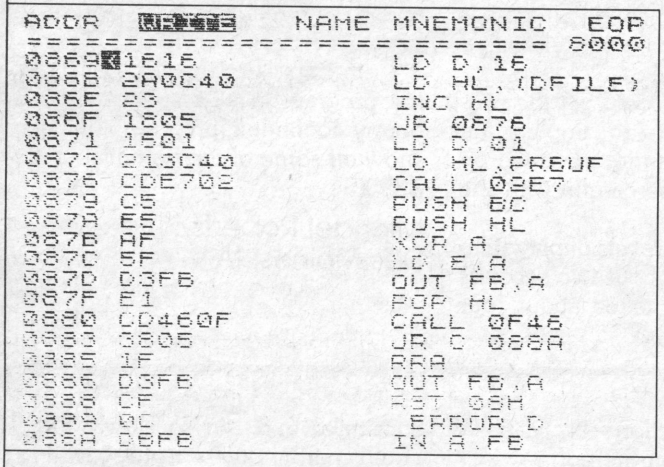
Hot Stuff

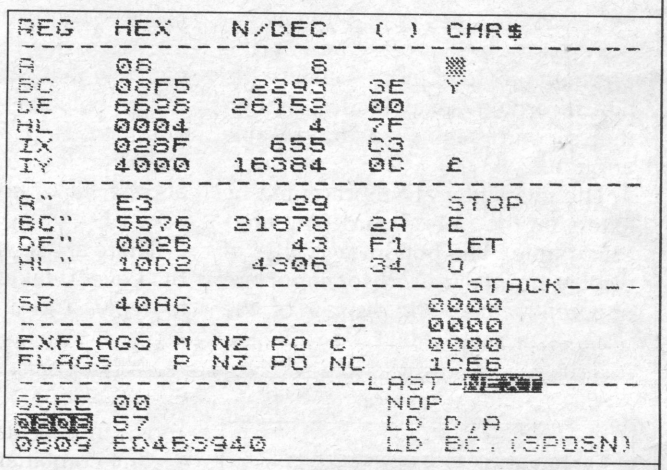
Name: HOT Z
Type: Utility
ROM/RAM required?: 8K/16K+
Written in: Machine Code
Listable?: Disassembles itself
Easy to load?: Yes
Printed listing?: No
Display: Excellent
From: SinWare
Box 323
Dixon, NM 87527
Price: $19.95 (Specify 16K or 32K)

Some programs have that professional aura that tells you as soon as you run them that your money, no matter how much the package cost, was well spent. HOT Z falls in this category.

Easily the best machine language debugging package I have ever seen for the ZX81 or TS1000, HOT Z is simple to use, well documented, intelligently written—as close to perfection in programming as has been achieved on the ZX81. HOT Z requires the SLOW mode, so it will not run on 8K ZX80s.

Imagine a window into the memory of your computer: one display that shows you assembly language listings of machine code programs stored in memory, along with hexa-




REG HEX N/DEC ( ) CHR$
A 08 8
BC 08F5 2293 3E Y
DE 6628 26152 00
HL 0004 4 7F
IX 028F 655 C3
IY 4000 16384 0C £
A' E3 -29 STOP
BC' 5576 21878 2A E
DE' 002B 43 F1 LET
HL' 10D2 4306 34 O
STACK
SP 40AC 0000
0000
EXFLAGS M NZ PO C 0000
FLAGS P NZ PO NC 1CE6
LAST-NEXT
55EE 00 NOP
0809 57 LD D,A
0809 ED4B3940 LD BC,(SPOSN)

decimal addresses, contents, and English-language labels; another that shows you all the registers plus a small window into memory where you may single-step through a machine code program; and another that shows data bytes in memory in hexadecimal, decimal, and character code form. Other features include a ZX81 floating-point calculator disassembler, decimal and hexadecimal entry of addresses, and a user-definable symbol table, so that you can read LD HL,(PRT1) in the disassembly instead of LD HL,(401DH). Some symbols, like DFILE, are permanently defined, and make understanding screen routines immeasurably easier.

SinWare provides an extensive and well-written manual with HOT Z. Although it assumes a familiarity with machine code and assembly lanuage, it is not too complicated for the beginner. It incorporates a long "introductory tour" of HOT Z which didn't appeal to me since I have used many disassemblers, and understand their capabilities. This section will be a big help to anyone unfamiliar with Z80 assembly language and architecture.

From using the package and reading the manual, I can tell it is well thought out and well written. Every command is called internally through a table of addresses, meaning you can change any function, or even just rename them if you want. There is space for your own commands, and one of the easiest ways of debugging a routine is to patch it in to HOT Z and then invoke it with a single keystroke.

Not much is left out of HOT Z, unless you have an unwavering need for an assembler: HOT Z is not an assembler. This turns out to be convenient, since typing in hex codes and watching the assembly language appear is much easier than typing in assembly language on the tiny keyboard of the ZX81. The system takes up most of 16K, but if you have 32K or more, one version of HOT Z will run above 16K.

If you program in machine language and need the best tool for the job, buy HOT Z. This program will never lose its value.

ZX/TS University

Program: Fun to Learn: English Literature 1 & 2, History 1, Inventions, Music, Geography
Type: Education
ROM/RAM required?: 8K/16K
Written in: BASIC
Listable?: Yes
Easy to load?: Yes
Printed listing?: No
Display: Good
From: Sinclair Research Ltd.
3 Sinclair Plaza
Nashua, NH 03061
Price: $12.95 each

Sinclair Research offers some exceptionally good educational programs in their Fun to Learn Series, written by Psion in England. Because of their UK origin, these programs emphasize English and European facts, particularly in History and Geography.

English Literature 1 includes two programs—one on novelists and one on authors. The computer organizes a race, pitting you against three other players to answer difficult literature-related questions.

Surprisingly, this program is not self-starting after loading. You must enter RUN manually. The first screen is a menu. You first choose among seven topics: Who Wrote This (one and two), Match These Works (three and four), Author—What is the Truth, Fact—Stranger than Fiction, and Potpourri (five, six and seven). To specify your choice, just hit a key from one to seven.

The competition is represented in the form of a line with you at one end as an X. Every time you answer correctly, the X advances toward the goal. When you answer incorrectly, you move back spaces while the computer gives you the correct answer.

English Literature 2 covers poets and playwrights and modern authors. It has the same topics and set-up as English Literature 1. Because both of these programs used some of the same data they reinforced learning. The questions are difficult though, and would be most appropriately used on the high school level or above.

History 1 covers events in British history and British monarchs. In the history program you choose a period in history from 1066 to 1981 and a mode: teaching (chronological), teaching (random order), events or dates.

In the teaching modes dates are displayed with selected events in chronological or random order until you press 0 to change the mode or hit the space key to abandon.

Events and dates modes set up a competition as in the literature programs. You are given six choices to match up an event with a date or a date with an event. Again, the computer provides the answer if you respond incorrectly.

Inventions are broken up into inventions before 1850 and after 1850 and is set up in the same competition form as the literature programs. You choose among seven topics: What Was Invented By. . .? (one and two); What Was This Inventor Like?; Identity Parade; What Do You Know About It?; Which Invention?; and Potpourri. This program, like the history program, would be best utilized in grades four or five and above.

The Music program, broken down into musicians and composers, is also more appropriate for high school level and above. It covers opera, musicians, composers, and a general sampling of other musical scores. Its format is the same as the Invention program but gives appropriate categories for music instruction.

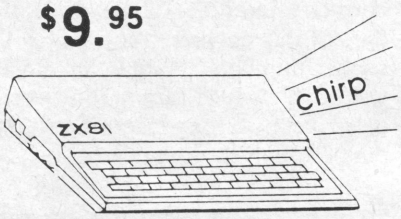
Geography presents some good graphics in the form of a map of Europe. The program runs in four modes: teaching (a country or capitol is named), select (choose the name of a country from a given list), Name the Country, Name the Capitol. In the teaching mode, the country or capitol is indicated on the map by a flashing question mark and you are queried for country or capitol.

The second program, Towns, shows you a map of England and Wales. A flashing cursor indicates the towns. There are three modes: teaching (it tells you the name of each town the computer selects), choice (you try to give the correct name of the town from ones the computer suggests) and name it (you try to name each town without any prompts).

Geography provides good graphics and prompts for learning and would be most appropriate for grade four and above.

These programs are easy to use, well developed and well prompted. Each would be an advantageous teaching tool in both classroom and home, but most would prove too difficult for young students.

$9.95



CHIRPER module for your ZX81 or your timex/sinclair 1000.

The CHIRPER module lets you enter keyboard data fast and accurately. A sound can be heard when a key has been entered enabling you to spot a double entry or missed entry without looking up at the screen.

The CHIRPER sound is produced when a program runs. A key entry results in a buzz-like chirp on 1K or 2K machines. Large programs in a RAM pack produce a continuous sound that chances pattern on key entry.

The CHIRPER module installs easily inside the ZX81 case with only 3 wires to connect. Complete installation instructions included.

Send a check or money order. We pay the postage in the U.S.A. or canada.

AUDIOGRAPH CO.
3584 Leroy, Ann Arbor, MI 48103

DATA BANK

The best software and support for your TS1000, Commodore and VIC, Atari, Apple, TRS-80 and . . .

Data Bank

394 East Campbell Ave.
Campbell, CA 95008
(408) 370-0199

3820 Peralta Blvd.
Fremont, CA 94536
(415) 790-1070

Memory with a Memory

Product: 8K Non-Volatile RAM
From: Hunter
1630 Forest Hills Drive
Okemos, MI 48864
Price: $29.95 for kit with 2K
$15 for board & manual only
add $1.95 (P&H)

What a super product! Hunter's non-volatile 8K RAM lets you program your ZX/TS from the keyboard, but unlike most RAM, retains your program when you turn off the computer. You can store a 2K program for 10 months and an 8K program for 2 months in this RAM with no external power. Paul Hunter conceived and executed this battery-backed RAM very nicely. Despite the low price, achieved in part by using jumpers instead of switches, leaving the unit uncased, and supplying only 2K with the starter board, the designer executed key properties carefully and with quality components. In addition, the instructions provide better-than-average (for ZX/TS accessory kits) guidance.

Most of all, the potential applications of this kit excite me. Since you can use a combination of EPROM and battery-backed RAM, you can create very flexible application modules with this device. It eases the way to dedicated ZX/TS applications by avoiding tape loading every time you restart.

Experienced kit builders will encounter little difficulty with this kit, though I found Hunter's estimate of two hours optimistic. I also feel that our circuit board vise eased the job considerably by holding the board conveniently.

When you finish the assembly, you own an uncased 8K memory board that you can position between 0 and 32K in 8K steps. As furnished, the board provides 2K of 6116 RAM and sockets for three additional RAMs or 2716 or 2732 EPROMS. You can add additional decoding (see SYNTAX newsletter, May 82) to position this memory above 32K. Remember that the system restart routine wipes out the contents of this RAM at addresses above 16K. With Hunter's decoding you cannot put the memory across a 16K boundary (as from 12-24K).

Hunter supplies a top quality board and connector and all the other components are also high quality. Color codes on two of the resistors (100K and 100 Ohms) are impossible to read, but an ohmmeter check identified those. You do not get a mating connector for the external battery—you'll need two sections of an Assembly Products 929974 or equal. Some builders simply solder wires from the external back-up battery to the board. I mounted the lithium battery on the circuit side of the board, insulated with electrical tape. This mounting really needs a more permanent insulator.

We made only the simplest tests—moving the 2K RAM from 8-10K up to 14-16K and moving a few bytes from one machine to another. Because the RAM is fully decoded, you can position the 2K anywhere within the 8K block chosen by jumper 1. Hunter's memory check routine works without problems, and I wrote a simple POKE and PEEK routine to store characters in non-volatile RAM and retrieve them. Using this routine, I moved data from one ZX/TS machine to another without problems.

Assembly Hints and Comments

Bending the pins on the edge connector requires a lot of force—you are bending 23 stiff wires at once. Use a piece of circuit board, metal or hardwood to bend the pins toward one another. When you finish, the pins should be less than 1/16 inch apart (less than the board thickness). Positioning of connector and card-edge extension are critical—if the solder joints on the extension go past the slot, your accessories will not fit properly. Also, the slot is too short, so 16K RAMs don't seat fully—put the RAM as far as it will go and keep it straight, don't let it rock around the key.

Leave service loops about 3/16 inch high on the jumpers so you can remove them for repairs or alterations. These loops, as well as C1 and Q1, must remain below the body of the edge connector or they'll hit the ZX/TS case when you fit the non-volatile RAM to the computer. Similarly, you probably want to take Hunter's advice and mount S1 on the circuit side of the board—you'll find it impractical to reach the front because your computer will block access. If you mount S1 at the rear, the 16K RAM clears it easily, leaving space for your finger to operate the reset button.

Do perform the tests suggested at various steps in the instructions. These procedures helped me find one bad solder joint and one that I missed (despite my 25 years of building such devices and an inspection after each group of soldering steps).

Don't be confused by the directions for D4—you use it differently for each of two independent decoding schemes—follow the instructions for the version you want to build.

Three minor glitches mar the instructions. One instruction refers to step 17, but the steps are not numbered, so you must count to find the correct reference. When you install the 24-pin sockets, they cover the socket numbers. Hunter fails to tell you which way to mount the external battery connector—insert the shorter pins through the circuit board.

At Your Fingertips

Product: Touch-A-Matic
From: KOPAK Creations Inc.,
Dept SY
448 West 55th Street
NY, NY 10019
Price: $9.95

After a slight (about two months) wait, I received a bulging envelope from KOPAK. It contained: a letter congratulating my patience, some product information, and a piece of thick, clear, flexible vinyl with forty holes spaced to the Sinclair keyboard.

With a heavy sigh, I decided to try it on my ZX80 first. Removing the white tabs that covered the adhesive, I tried to center the Touch-A-Matic

per the instructions, but it wouldn't quite fit. Ignoring this for the moment, I tried to enter a short program. The thickness of the vinyl made it hard to make the proper contact with the keyboard. It didn't increase my speed because I had to slow down to press harder.

Thinking that the Touch-A-Matic was made with the ZX81 in mind, I transferred it to my other machine. The fit is a little better, but still the vinyl did more to slow my program entering than to speed it. Even after two months use, I have yet to achieve the touchtyping as advertised.

On the whole, I have little respect for the Touch-A-Matic and can't recommend its purchase. Perhaps persons with smaller, nimbler fingers would be more successful.

Sgt. Patrick Spera,
Tinker AFB, OK

Hi Res Solution

Product: Memotech Hi-Res Graphics
From: Memotech Corporation
7550 West Yale Ave.
Denver, Colorado 80227
1-800-662-0949
Price: $144.95

Yesterday I received my Memotech high resolution graphics module. As usual with all Memotech products it comes in a black anodized case and piggybacks between the ZX81 edge connector and your RAM pack without a motherboard. Used in conjunction with their 64K Memopak, it gives a professional appearance to your ZX81. And since they were built to fit together, they are immune to the "wobble and die" syndrome.

To use the HRG, first set your Memopak to Mode B or D as the HRG uses the addresses from 8192-10240 (2000h-2800h), and remove the jumper from the HRG which sets the video page address above 40000 (9C40h). If you are not using a Memopak and are using a 16K RAM, leave in the jumper to set the video page at 19000 (4A38h).

It's important to read the directions because the HRG tends to be complex until you are familiar with the instruction format.

With all commands you must assign V = (video page address), and Z$ = (command). Other commands ease game applications like "Sketch" (user defined graphics) and "Launch", which fires a line until it hits something. The 30 HRG commands give complete control of the screen. You can save video pages in memory depending on memory size as each page uses 6337 bytes.

Advantages include: Top of the line Hi-Res screen control, fully decoded addresses (uses only 8-12K leaving free 12-16K), easy set-up and testing, user defined graphics, as well as ability to save and load video pages to and from cassette.

Disadvantages include: Non standard Hi-Res commands, slow graphics, lack of Hi-Res commands like circle, box, or paint.

Comparison and Evaluation

Before the arrival of my HRG I used a Quik-Silva Hi-Res Board. Because it is not fully decoded I had to shut off the 8-16K area even though its ROM is only 10-12K. With the HRG I reclaimed the 12-16K area gaining 4K RAM. If you currently have an accessory which uses the 8-16K area, if its address is not fully decoded or uses the 8-11K area, you can not use it with either the HRG or QS boards. To get full advantage of either product you need a monitor.

These two products differ in speed and flexibility. The QS Board is about three times faster. Yet the QS Hi-Res has only 12 commands versus HRG's 30. The HRG allows multiple video pages and easy MIC access to those pages and the QS does not. I chose the HRG for flexibility because control over your screen is amazing, although slow. When used in conjunction with the 64K Memopak it also allows you to use the 12-16K area for machine code subroutines.

Jeff Pack, Auburn, WA

Ephemeris V

PLANET FINDER FOR THE 16K TS-1000/ZX-81

ENTER: date, time, latitude and longitude. Then choose one of the principal planets, the Sun or the Moon. In seconds Ephemeris V responds with:
1. Azimuth and Altitude
2. Right Ascension and Declination (for astronomers)
3. Simple compass coordinates (for beginners)
4. GRAPHIC DISPLAY of planet in current constellation
5. LOCAL sidereal (star) time to aid in finding stars, comets, nebulae, etc.
6. The phase of the Moon.

Own or give this incredible program and reap the harvest of real understanding forever! Comes on quality cassette with instructions. Send a money order or check for $8.00 ppd to:

EPHEMERIS V
P.O. Box 261
Winchester, KY 40391

2K POWER! for 16K too!

GOBBLE 2K
== ==
A Fascinatingly Addictive Game for 1 to 4 Players. Nine Levels of Play, from Easy to Impossible. Each Player can Choose a New Level Each Turn. Loads of Fun and Excitement! $8.95

EZ-CHEK 2K
== ==
Easy to Use. A Fast-Loading Electronic Checkbook that uses Data-Compression Technique to store Details of 51 Transactions. Current Balance. $6.95

Add $1.50 per order p.& h.
NY residents add Sales Tax.

FLINT DATA SYSTEMS
8 Daisy Lane, Commack, NY 11725

VOTEM Lets Your Computer See The World

Analog Interface/Tape Signal Conditioner
(for the Timex/Sinclair Computer)

VOTEM is a complete package consisting of hardware and software that enables your computer to measure, display and record "real world" analog signals. Your computer can monitor any physical phenomenon (pressure, light, temperature, etc.) represented by a DC voltage. A probe is provided for air and liquid temperature measurements.

Your computer becomes a "smart" digital voltmeter and thermometer with storage capability. Use VOTEM in a home energy conservation project to save money and possibly qualify for an energy tax credit.

VOTEM also cleans up the tape signal for reliable LOADS. With VOTEM you can LOAD tapes with a lower volume setting on your tape recorder.

VOTEM requires no mods, does not use the computer's rear expansion connector and works with 1K RAM (or more).

VOTEM is low priced at only $59.95 (assembled and tested). For an even better bargain the VOTEM kit is only $39.95. (Requires soldering and appx 2 hrs.) VOTEM comes with a detailed 35-page manual. The manual may be purchased separately for $5 pp and applied to first purchase of a VOTEM unit. If you are not satisfied with VOTEM return within 15 days for full refund. (Does not apply to kits). Send check or money order plus $3 for shipping and handling.

Down East Computers
P.O. Box 3096/Greenville, N.C. 27834

Ad Index

Amusement Co. 48
Audiograph 61
Binary Corp. 48
Brainchild C2
Byte-Back 7
Capacity Plus 50
CED Corp. 64
CHD Productions 55
Computa-dek 4
Compute Magazine 55
Computer Continuum 64
Computer Ideas 50
Data Bank 61
Data Domain 45
Down East Computers 63
Empire Systems 50
Ephemeris V 63
Ezra Group II 19
Flint Data 63
Florida Creations 64
General Systems 55
Intercomputer 55
Kennedy Systems 48
Kopak Creations C4
Maryland Book Exchange 45
Memotech 1
Mindware 46
Monmouth Computer 48
Omnifax Computer 50
PC Magazine C3
Reston Publishing 15
Rist Computer 19
Softsync 2
Software City 19
Synchronized 4
Synware 4
Voice Tec 53

Computer Continuum

301—16 Ave. San Francisco
CA, 94118. (415) 752-6294

Peripherals for Serious Sinclair Users

Buffered Buss/Development Board
With full bi-directional buffering.
Widely accepted. .... **KIT $65**
BARE BOARD w/connector **$40**

UHF MODULATOR replaces VHF.
Ends interference problems . **$15**

RIBBON CABLE HARDWARE
Connector w/fingers
to fit peripherals **$7**
Connector to ZX/TS1000 ... **$6**

ANALOG INTERFACE
FAST! **8 Ch*(A/D+D/A)** NEW

Please add $3 for shipping charges.
Send stamp for complete details.
CAL res. please add 6% tax.

FLORIDA CREATIONS PRESENTS:

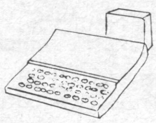
SINCLAIR SOFTWARE ★ CHEAP ★

» GAMES

» HOME

» BUSINESS

» TECHNICAL ARTICLES



ZX-81 and TS 1000
1k through 16k memory
MOST PROGRAMS 50 CENTS EACH OR LESS

Catalog & two sample programs for $1 and SASE

FLORIDA CREATIONS, DEPT. SQ1, P.O. BOX 16422, JACKSONVILLE, FL, 32245

****SATISFACTION GUARANTEED*******

Software by C.E.D. CORPORATION

E-Z Bookkeeping for Small Business
(takes less than ½ hour per month).
Good for Budget-Minded Homemakers

Checking Account
(makes child's play of this job)
By itself it's great but it was designed to work with E-Z Bookkeeping

Machine Language Nit-Pick
(great for learning assembly language programming)

Word Processor

Order from

C.E.D. CORPORATION
P.O. BOX 222
SALEM, WISCONSIN 53168

Price List

	TAPE	LIST	BOTH
E-Z Bookkeeping	$19.95	$8.95	$27.95
Checking Account	$9.95	$5.95	$14.95
Above Two as a Package With List			$41.95
M.L. Nit-Pick	$18.95	$7.95	$25.95
Word Processor	$18.95	$7.95	$25.95

All of our programs are menu driven and all data can be stored on tape and/or sent to a line printer.

We have telephone support for problems or questions:

(414) 843-3990 **(414) 276-8787**

SYNTAX QUARTERLY
by The Harvard Group

Increase Your Computer's Utility With Our Help

You own a powerful computer, capable of sorting, analyzing, calculating, displaying and manipulating data as well as measuring and controlling. Despite its small size, your computer will help you learn, analyze your business, keep your records or your schedule, dial your phone, send messages, or talk to other computers. To control the power available to you, you'll need hardware and software to use with your machine. Although you can buy many of these products—and we'll tell you about them—some you must create or modify for yourself. SQ will provide you with the complete information to let you use your programming or tinkering time efficiently.

Our experts also bring you SYNTAX, the newsletter, for up-to-date, concise information. You need news and new product announcements quickly. And SYNTAX packs a lot of information into its brief, time-saving format.

```
100 LET A$="█████████"
200 LET B$="█       █"
250 LET C$="█"
300 PRINT A$,A$
400 PRINT B$,B$
500 PRINT B$( TO 7),B$
600 PRINT B$( TO 7),B$
700 PRINT A$,B$
800 PRINT " ";B$(2 TO ),B$
900 PRINT " ";B$(2 TO ),B$
1000 PRINT B$,B$
1100 PRINT A$,A$
1200 PRINT AT 7,22;C$
1300 PRINT AT 9,24;C$
1400 FOR I=0 TO 248
1450 IF I>66 AND I<128 THEN GOTO
 1600
1500 PRINT CHR$ I;
1600 NEXT I
1700 PRINT AT 10,0;"  WE SPEAK Y
OUR LANGUAGE
SYNTACTIC SUM:  22174, 8K ROM
```

Interested in the IBM Personal Computer?

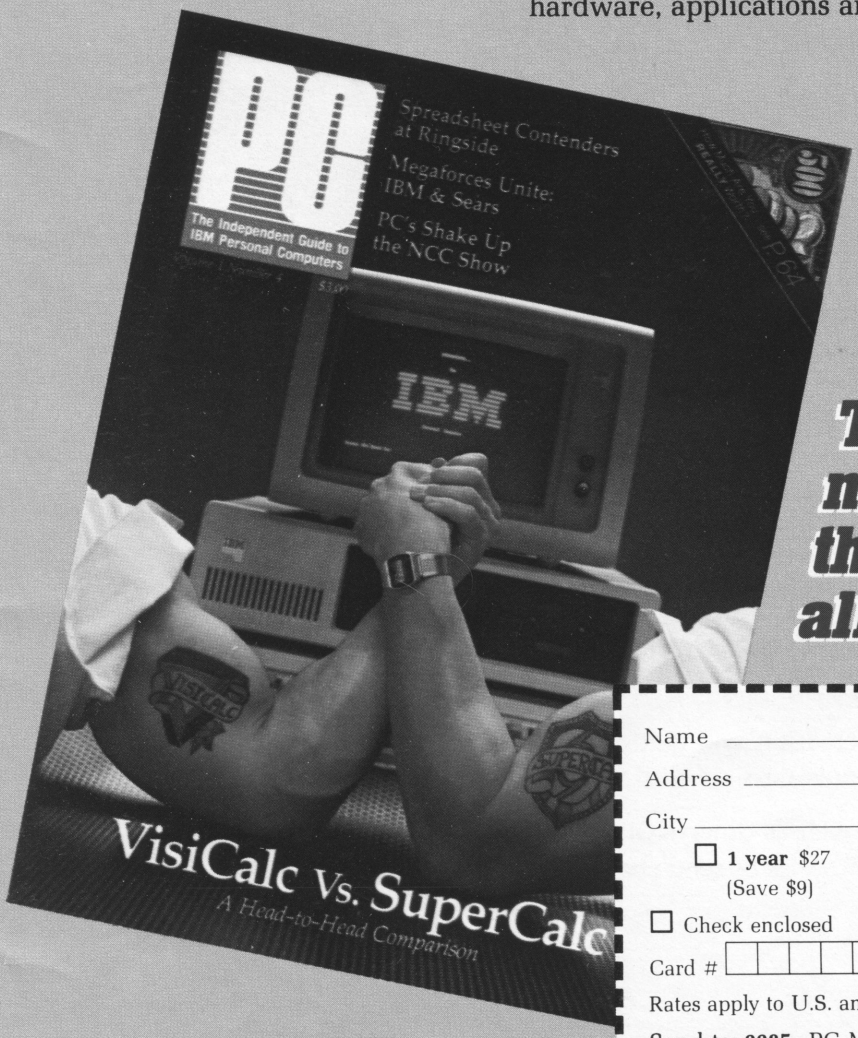
If you're interested in the IBM Personal Computer then you need PC magazine.

PC magazine is the Independent Guide to IBM Personal Computers. Each issue is packed with information for everyone interested in IBM Personal Computers.

PC magazine tells you how to put together the best IBM "PC" system and then how to get the most out of it. Each issue brings you hundreds of colorful pages of evaluations, insights, and straight talk from respected experts—professionals in computer science as well as writers, businessmen, lawyers, educators, and many others.

PC covers software, hardware, applications and most every topic of importance to the thousands of IBM Personal Computer users who read it. To ensure that we give you the information you need, PC includes a special "User-to-User" section, as well as a "PC Wish List", and news about IBM Personal Computer clubs, events and publications.

For a limited time, you can subscribe to PC at NO RISK and still receive a 25% discount off the newsstand price. Enter your subscription now. If not fully satisfied when you receive your first copy of PC, simply return your mailing label within 15 days for a full refund.



This is the magazine that tells you all about it.

PC
The Independent Guide to IBM Personal Computers

Name _____

Address _____

City _____ State _____ Zip _____

☐ **1 year** $27 (Save $9) ☐ **2 years,** $46 (Save $26) ☐ **3 years,** $59 (Save $49)

☐ Check enclosed ☐ VISA ☐ MasterCard ☐ Bill me

Card # _____ expires _____

Rates apply to U.S. and Canada only. All others double.

Send to: **0085** , PC Magazine
1528 Irving St.,
S.F., CA 94122

Phone Credit Card Orders to:
(Toll Free Number)
California: 800/792-0990, ext. 1136
All Other States: 800/227-3800, ext. 1136

"Touch-A-Matic™", Power At Your Finger Tips!

$9.95

KOPAK'S TOUCH-A-MATIC™ gives you the power to type more accurately and much faster. It requires no wires, no soldering. Comes with complete instructions. It's as easy as removing adhesive backing and pressing into position. Positioning is easy. Once in position, you are ready for touch-typing with ease.

Our unique vinyl-key-hold creation will guide your fingers to the correct keys. Finally, touch-typing now possible with your Sinclair* or Micro-Ace*.

This remarkable product, as well as other **KOPAK** items, are now available. Call now to order through MC/Visa or send check/M.O. to **Kopak Creations, Inc.**

TM Trademark of KOPAK Creations, Inc.
Sinclair* is a trademark of Sinclair* Research LTD.
MicroAce* is a trademark of MicroAce*

KOPAK CREATIONS, INC.
(212) 757-8698
Master Charge & Visa Accepted

$1.50 Handling Charge

KOPAK Creations, Inc. SQ1 **448 West 55th Street, New York, NY 10019 (212) 757-8698**